Dissertation

# Interactive Multiagent Adaptation of Individual Classification Models for Decision Support

Tobias Warden

zur Erlangung des Grades eines
Doktors der Ingenieurwissenschaften
– Dr.-Ing. –

Vorgelegt im Fachbereich 3 (Mathematik / Informatik)
der Universität Bremen
im Februar 2019

| | |
|---|---|
| Gutachter | Prof. Dr. Otthein Herzog<br>Universität Bremen |
| | Prof. Dr. Winfried Lamersdorf<br>Universität Hamburg |
| Disputation | Bremen, den 02.07.2019 |

## Dedication

*For my wife Fatiha, my son Yassin, and my loving parents.*

# Acknowledgements

I would like to express heartfelt gratitude to my family, first and foremost my parents, and my wonderful wife Fatiha, my thesis supervisor Professor Herzog, as well as good friends and colleagues, chief among them Sandra Budde, Christian Ober-Blöbaum and Christoph Greulich. They all have provided invaluable support and have displayed unwavering trust over the whole course of this doctoral dissertation project.

Where I faltered in the face of the challenges encountered, they have helped me in their own respective ways at various stages of this project. These stages include the initial definition of the research scope which they have helped to narrow down with feedback in long discussions. Hereafter, the related work research and initial system design, where I have benefited from the expertise of friends and colleagues. The next stage involved experiment design and implementation, all in the face of an ending employment in the Artificial Intelligence working group. As this state of affairs rushed the compilation of the first drafts of this document, I would have given in at this point without the encouragement of family and friends. They also helped me to keep going even when I went on from the well-known world of academia to explore software development in the fast-changing area of e-Commerce.

It was at this time that I had the privilege to get to know my wonderful wife. She was responsible to push me to assemble the rough pieces of my work, proceed to refine them into a coherent whole and thus conclude a long journey.

I would like to especially to thank Professor Herzog for his support and feedback throughout this project and the encouragement and patience while waiting for its eventual completion.

# Contents

Chapter 1

# Introduction

Multi-agent technology has been adopted in recent years for applications in the logistics domain by a growing number of research groups (Henesey et al. 2008, Davidsson et al. 2008). The bandwidth of applications includes for instance the management of manufacturing and assembly processes in production logistics (Lorenzen et al. 2006), business-to-business coordination in the context of supply chain management (Chaib-draa & Müller 2006), operative transport planning (Bloos et al. 2011, Wojtusiak et al. 2012*b*, Pantke 2017) and optimised last-mile distribution to end-customers (Gath 2016). Software agents in these application contexts have come to represent either logistic service providers or individual human decision-makers, such as production managers or dispatchers, that have traditionally controlled logistic processes in their companies. Multi-agent technology has thus been applied as a means for automation and optimisation of existing processes.

The paradigm of autonomous logistic processes has gone beyond such applications as the control of logistic processes has been delegated to individual logistic objects themselves (Böse & Windt 2007). Autonomous logistic entities that are represented by software agents plan and supervise their own passage through logistic networks. Research in the field has initially focussed on the collaboration and coordination of individual agents that is critical to the fulfilment of logistic objectives (Schuldt 2011, Berndt 2016).

This doctoral research complements previous research on multiagent collaboration for logistic processes with a focus beyond immediate process control. Collaboration is investigated in the context of agent-oriented knowledge management, specifically individual learning of classification models in an agent society that enable informed decision-making in process control.

As constituents of multiagent systems for autonomous control of logistic processes, software agents can be characterized by their respective operative roles, such as the management of logistic resources (Langer et al. 2007). In transport logistics, such resources encompass amongst others means of transport or handling equipment, as well as storage facilities (Warden & Wojtusiak 2010, Wojtusiak et al. 2012*b*). In production logistics, managed resources are rather machines or material handling equipment in assembly lines. Other roles involve the management of the subjects

of the logistic functions themselves; for instance work pieces (Ganji et al. 2010) and commodities (Schuldt 2011).

Within the scope of their respective roles, agents assume responsibility to execute logistic tasks on behalf of their attributed logistic entities. Thus, rational and informed decision-making subject to these roles is the primary goal of the agents.

## 1.1 Problem Definition

An essential prerequisite for informed decision-making is direct access to, or the ability for a utility-driven situation-sensitive retrieval of *adequate information* (Gehrke 2011) and *empirical knowledge* for situation assessment (Gehrke & Wojtusiak 2008*b*, Wojtusiak et al. 2012*b*). To acquire both, agents assume *auxiliary roles* which implement an agent-oriented knowledge management in addition to their primary domain roles (Langer et al. 2007). Knowledge management is qualified here as agent-oriented to highlight that, in contrast to traditional use cases (Smirnov et al. 2002), both providers and consumers of knowledge management functions are software agents.

Besides shared a-priori domain knowledge, modelled formally by means of ontologies (Warden et al. 2010, Porzel & Warden 2010), a vital form of knowledge to be provided via dedicated roles, especially in dynamic environments, is empirical knowledge. Jennex has characterised such dynamic knowledge as information "*understood such that it explains the how and the why about something*" (Jennex 2009), i.e., knowledge that is applicable to situation assessment and prediction. Dynamic knowledge subsumes several categories of integral knowledge models for decision making, including decision models, prediction models, and classification models.

Examples in freight haulage and supply chain management include models for traffic densities within a transport network (Gehrke & Wojtusiak 2008*b*,*a*), prediction models for handling time at transshipment points, or delays of shipped orders. In fabrication and assembly, relevant predictors allow, for instance, for the estimation of waiting and processing times at machines along assembly lines (Scholz-Reiter et al. 2008).

The knowledge models required for an individual agent scope ensue from the primary roles of the agent. The requisite on dedicated auxiliary knowledge management roles are hence described as follows: Once an agent is to assume a new primary role, the therefore required knowledge models with acceptable predictive performance as measured in terms of metrics such as prediction and recall need to be rendered available in a timely fashion.

This research project thereby assumes that the aforementioned models are constructed individually by each agent by means of machine learning on its personal empirical data.

Following the characterisation of Kazakov & Kudenko (2001) for machine learning in the context of multiagent systems, the agents perform supervised single-agent learning. Whereas traditional machine learning separates the actual learning from the acquisition of a representative body of empirical data to construct

training sets as two independent processes, software agents need to perform and integrate both processes. Their situation is related to the one found in closed loop machine learning in that the agents gather empirical data in a goal-directed way to support further learning.

Due to the assignment of primary roles, the agents need to learn on the job. Thus, the empirical data which is required to learn a prediction model to support decision-making in the context of a primary role is acquired from the observation of the situation context of the agent while already assuming the aforementioned role.

This situation bears implications for the agent learning tasks: First, the empirical data that can be acquired is biased by primary roles, their respective specialisation, and the environment (e.g., due to a particular assignment of haulage or production orders). Second, since agents already need to perform their operative roles competitively while, in the background, learning models to support decision-making, a prompt availability of those models is critical. Specifically, it may be infeasible for the agents to first acquire months or even years worth of empirical data as assumed in learning experiments by Gehrke & Wojtusiak (2008*b*,a) for learning prediction models for traffic densities.

The argument holds especially for learning in *dynamic environments* with concept drift (Zliobaite 2009) where concepts change due to seasonal or economical effects which are not factored into the trained models.

When in the context of an auxiliary knowledge management role, an agent fails to learn a classification model with acceptable prediction quality – implying that its training set has not been representative – its only option for *self-sufficient* action is the acquisition of further empirical data. However, this approach may take considerable time. Furthermore, due to the constraint that the agent acts in order to best serve its primary rather than its knowledge management objectives, it is hard if not infeasible to actively explore the environment to effectuate making 'helpful' observations. Consequently, there is a need to establish novel options for the individual learner to enhance the basis for its learning processes.

To that end, the following observations from practical implementation of multiagent systems controlling logistic processes (Gehrke & Wojtusiak 2008*a*, Wojtusiak et al. 2011) apply: Depending on the granularity of autonomous control, the primary agent roles in particular processes may not be performed exclusively by single agents, but by groups of agents. The decentralised operative transport planning for individual trucks in a freight forwarder fleet investigated in (Wojtusiak et al. 2011) constitutes a paradigmatic example: In this scenario, several agents of a logistic service provider assume transport management roles. Thereby, each agent acts on behalf of a single truck. The decision-making for such primary roles relies on a common set of dedicated knowledge models; for the planning of transport routes, these can comprise prediction models for trans-loading times or traffic flows in the transport network (Gehrke & Wojtusiak 2008*a*). Even when distinct role specialisations exist[1], a close relation within a taxonomy of logistic roles may still imply the employment of similar knowledge models. Each of the related knowledge models is thereby based on empirical data of the respective

1 For instance, pre- and onward-carriage versus main leg, or transport of regular goods vs. transport of hazardous goods (Vahrenkamp 2007).

agent. It has been acquired in specific operative contexts characterized amongst others by their spatial and temporal dimension. For instance, the local delivery area of a freight forwarder may be split into (overlapping) delivery areas.

Autonomous control of logistic processes has therefore constituted *one paradigmatic instance* of a system environment in an application domain where groups of agents within a multiagent system each face kindred single-agent learning tasks to construct situated prediction models in order to support *identical* or *related* primary roles that are played in *overlapping* operative contexts.

The problem definition which ensues is then to unlock potentials for the enhancement of individual learning in system environments exhibiting the aforementioned characteristics.

## 1.2 Research Hypothesis and Approach

Derived from the problem definition in the preceding section, the central research hypothesis for this doctoral research can be stated as follows.

> "*Under qualified conditions, both the predictive accuracy and appropriation time including data acquisition of individual dynamic knowledge used in decision making can be optimised when the learning agent is equipped with effective means to leverage empirical dynamic knowledge of suitable non-adversarial interaction partners in a multiagent system.*
>
> *This goal can be accomplished in a distributed approach to knowledge transfer which retains the agents' responsibility for their own learning tasks and does not require the centralisation of empirical data.*"

The presented research claims that for those agents not acting in isolation, the multiagent environment is quintessential to complement individual adaptation capabilities of an intelligent agent. Let a persistent multiagent system host multiple agents with primary domain tasks of similar type and associated supporting models for decision making. These models may contain knowledge that is beneficial for an agent that has self-sufficiently experienced an adaptation impediment.

At its core, the fundamental idea of this research is to furnish software agents, which are already capable of local learning from their own body of experience, with a complementary ability to leverage forms of cooperative social learning for the acquisition of dynamic knowledge. This new learning modality thereby revolves around the requirement-driven transfer and subsequent internalisation of knowledge from cooperating peers within a common community of practice.

The surplus value from the point of view of the individual agents lies in the potential to efficiently mitigate weaknesses in their respective knowledge models. In particular, this holds for situations where it is too tedious or time-consuming to work out model deficiencies using self-sufficient local learning alone. The rationale is that agents now have the option to learn from peers and thus benefit from their previous learning efforts.

From a global point of view, the desired effect of the introduction of agent-oriented knowledge management and its continued effectuation through many individual knowledge transfer episodes lies in an emergent improvement of dynamic knowledge that is in operative use within the agent community.

An additional incentive for the development of successful strategies for collaborative knowledge management is the perpetuation of proved and tested knowledge, be it complete decision support models or just certain aspects thereof. The transmission of such knowledge is of particular relevance in persistent multi-agent systems with a steady turnover of active agents. Essentially, the formerly tightly catenated life cycles of agents and their personalised knowledge models become decoupled. Thus, the knowledge which constitutes a valuable corporate asset is stored both decentralised and redundantly in the agent community such that even in the face of, for instance, an unforeseen termination of important knowledge bearers, the gist of their knowledge is still retained.

Consequently, this thesis proposes a practical agent-oriented knowledge management framework that enables dynamic knowledge networks. It facilitates agents overcoming the inhibiting factors for individual adaptation by drawing on appropriate knowledge distributed among peers in a goal-oriented way. Key to the proposed framework are dedicated knowledge management functions that are implemented by agents seeking a value-added adaptation of individual classification models supporting decision making and their supporting counterparts.

It is acknowledged that dynamic knowledge may not be transferred among agents simply by creating carbon copies due to heterogeneity in the respective knowledge representations. Hence, a discourse based, interactive adaptation process envisions those agents seeking to engage in aided adaptation to actively acquire, interpret and process advice from peers. In doing so, the advisees selectively reconstruct useful aspects of dynamic knowledge that advising agents codified in their own models.

The approach to address adaptation impediments through social interaction conceptually emulates human problem solving strategies and confers aspects thereof upon intelligent agents.

## 1.3 Research Context and Contributions

The research presented in this thesis has been inspired by preliminary work conducted within the Collaborative Research Centre 637 "*Autonomous Cooperating Logistic Processes - A Paradigm Shift and its Limitations*" (Freitag et al. 2004) over an eight-year term.

It approaches the field of interacting multi-agent learning for a sustainable knowledge transfer within multiagent systems from an integrated perspective which factors in some input from the fields of distributed artificial intelligence, knowledge management and machine learning. The contribution of this work can be broken down into four main themes:

**Contribution 1** It develops a concept for discourse-based adaptation of individual dynamic knowledge, specifically classification models, building on extensive interaction with knowledgeable peers within a multiagent system. The adaption problem is modelled as local search problem in the space of learning hypotheses reachable by iterative integration of learning advice from interaction partners. Existing work on interacting learning is incorporated and refined for use in multiagent environments.

**Contribution 2** It contributes integral parts to enable the goal-oriented formation of dynamic knowledge networks. This comprises mechanisms for discoverability and enlisting of agents as knowledgeable interaction partners, i.e., experts for a specific learning task.

**Contribution 3** It integrates the concept of discourse-based adaptation of local decision support models into a complete agent architecture such that the adaptation processes of the agent are kept transparent from the performance element's point of view. The agent is equipped with meta-reasoning capabilities for a continuous assessment of operationalised models that determine the scheduling of discourse-based adaptation activities.

**Contribution 4** It contributes an extensive test environment for the empirical assessment of this integrated adaptation approach with different compositions of agents and learning input. The test environment is implemented on top of the general-purpose multiagent-based simulation environment PlaSMA.

The research follows a constructive approach in that the focal point is on the investigation of artificial intelligence techniques to provide the aforementioned instruments as enablers for agent-oriented knowledge management and consequently improved learning behaviour. The intentional constraint to investigate a distributed approach is thereby geared towards characteristics of autonomous control. The research is also analytic as it will investigate not only the potential but also the limits of the investigated methodology. For instance, it should be investigated if specific combinations of learners or empirical models are essential for success. Finally, the doctoral research bears a practical orientation. Its goal is the provision of central building blocks to enable collaboration in agent-oriented knowledge management in multiagent systems.

## 1.4 Thesis Structure

To address the outlined research hypothesis, this thesis is divided into three parts. The first part is dedicated to the investigation of relevant prior research from three complementary research fields, namely intelligent agents and multiagent systems, knowledge management from an agent-oriented point of view, and finally machine learning and its application to multiagent environments. The second part of the thesis merges concepts from these fields and integrates them to enable the

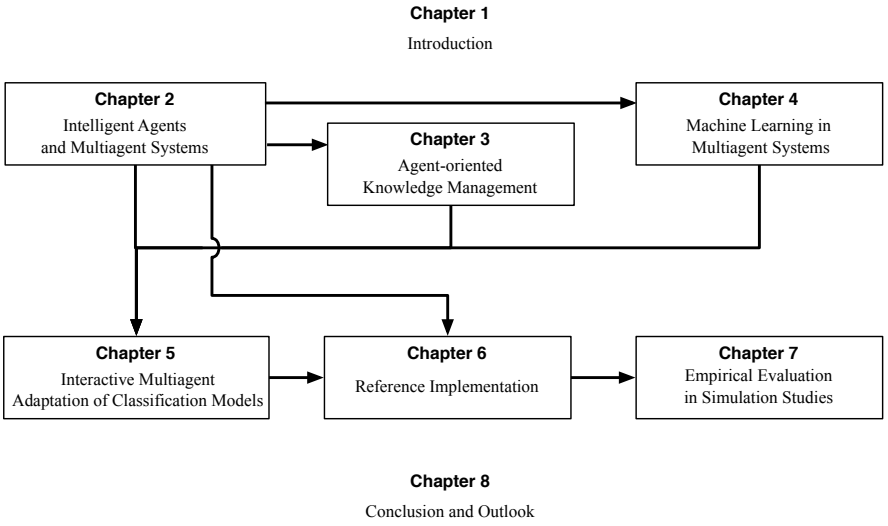



**Figure 1.1:** Overview of the thesis structure. Major interrelationships between chapters are indicated by arrows and explained in the text.

discourse-based adaption of individual classification models for decision support. This part comprises the major contribution of this thesis. The third and final part is dedicated to the concrete implementation of the developed concepts in a multiagent system and an extensive evaluation by means of multiagent-based simulation. The structure of the chapters is illustrated in Figure 1.1. Their interrelationship is as follows.

### Chapter 2 · Intelligent Agents and Multiagent Systems

To begin with, fundamentals of intelligent agents are introduced with a particular focus on knowledge as an enabler of sophisticated agent architectures. Acknowledging the importance of knowledge for an agent performance element, the consequential incorporation of learning capabilities and hence the transition to learning agents is discussed. The necessity and architectural framework for individual agents autonomously shaping the evolution of their own knowledge base over the course of their life cycle is hence established. As learning agents are often embedded in an environment with other knowledge-based or even learning peers, the means to enable rich conversations among these agents on the subject of information exchange for individual knowledge furthering in standardised agent environments are presented. Finally, multiagent-based simulation is introduced as an essential means for an effective study of dynamic knowledge networks involving multiple learning agents in carefully controlled situational contexts.

### Chapter 3 · Agent-oriented Knowledge Management

Based on the established premise of knowledge distributed or rather scattered among individual agents in agent societies, knowledge management is explored as an interdisciplinary field interested in developing means to facilitate the transfer of knowledge among individuals, aiming at a continuous improvement process and

knowledge permeation. An introduction of knowledge management fundamentals serves to render more precise notions of knowledge and constituent processes driving knowledge management. An outline of knowledge management systems as the discipline matured establishes the growing involvement of intelligent agents and even multiagent systems as digital representatives facilitating the participation in knowledge management for human stakeholders. Finally, based on the outset of this thesis and the focus on learning agents in the preceding chapter, it is claimed that a paradigm shift in the understanding of agent technology as mere means for advanced human-oriented knowledge management is necessary. Consequently, agent-oriented knowledge management acknowledges the learning agents themselves as the essential stakeholders and subject of knowledge management activities.

**Chapter 4 · Machine Learning in Multiagent Systems**

Machine learning contributes the essential building stone for the implementation of learning agents following the architecture from Chapter 2. Hence, a landscape of machine learning methods with a focus on classification precedes an exploration of the theme of autonomy to shape learning processes in the meta-control of advanced machine learning systems that are researched in active learning and related approaches. It is shown that, at least with respect to learning activities, the transition from learning system to learning agent is fluent. In terms of Chapter 3, the crucial finding is that learning systems are enabled to actively reach out to human domain experts in knowledge acquisition. While the machine learning community explores learners that adopt traits of agency, learning is also explored specifically in multiagent systems. Interacting learning is identified as the approach that suits the requirements in this thesis.

**Chapter 5 · Interactive Multiagent Adaptation of Classification Models**

This chapter finally integrates the findings and contributions from the fields of agent technology (Chapter 2), agent-oriented knowledge management (Chapter 3) and interacting learning by means of argument based machine learning as a form of discourse-based active learning (Chapter 4) to synthesise a novel concept for the multiagent interactive adaptation of classification models for decision support. The contribution is an integrated concept that fleshes out the paradigm of agent-oriented knowledge management. The chapter involves a systematic approach based on an initial collection of requirements, the development of necessary knowledge management roles in order to establish and sustain dynamic knowledge networks in multiagent systems, a formalisation of interactive classification model adaptation as an online search problem, and finally the specification of all building blocks whose coaction effectuates the desired autonomous knowledge adaptation in multiagent systems.

**Chapter 6 · Reference Implementation**

The paradigmatic implementation of interactive multiagent adaptation of classi-

fication models developed in the preceding chapter is finally presented in this chapter. It introduces a flexible methodology for the implementation of knowledge management roles based on finite state machine behaviours as advocated by the leading FIPA-compliant multiagent platform JADE. This methodology affords a flexible coupling of agents and knowledge management roles based on capabilities. Thus, role assignments are dynamically based on trigger conditions and not predetermined by the agent engineer at design time.

**Chapter 7 · Empirical Evaluation by Simulation Studies**

To study the performance of the implemented multiagent system, simulation as introduced in Chapter 2 is employed in the evaluation. To that end, an important contribution is the extension of the multiagent-based simulation system PLASMA with an infrastructure to support highly-configurable learning experiments with any number of learning agents. A comprehensive empirical study of the concept proposed in Chapter 5 by means of multiagent-based simulation in two application use cases is finally documented. It confirms the validity of the proposed concept and offers both a macro evaluation of accumulated results across experiment series in terms of a suitable system of key performance indicators, and a micro evaluation which examines specific adaptation episodes.

**Chapter 8 · Conclusions and Future Work**

The report closes with a conclusion in which initial research questions are revisited in the light of the findings acquired in the preceding chapters, especially the previous evaluation. This leads to the identification of directions for future research, i.e., a systematic further development of the proposed contribution to agent-oriented knowledge management touching on critical points such as the increase in heterogeneity in classification approaches employed in investigated multiagent systems, or the topic of semantic interoperability.

Chapter 2

# Intelligent Agents and Multiagent Systems

Coincidental with the maturation of agent technology, recent years have seen a broad adoption of architectural paradigms that favour flexible ensembles of autonomous, intelligent, knowledge-based software entities with local problem-solving capabilities and the potential for coordinated action in order to reach both individual and joint objectives (Kirn et al. 2006, p. V). Rather than solving complex problems with an inherent spatial, organisational, and logical distribution by means of traditional centralised control, decision-making authority is being delegated to digital representatives of previously inanimate process stakeholders.

This theme of distributed control is well-known in multi-robot domains as exemplified by the scenarios investigated in the RoboCup initiative (Visser & Burkhard 2007). It was also the motivation for the paradigm of autonomous logistic processes (Freitag et al. 2004) and its recent successor, Cyber Physical Systems and the industry 4.0 initiative (Lee et al. 2015) that seeks to extend the process of digitalisation for the complete production and logistics business processes by means of Cyber Physical Systems to the industry sector. A driver for the advance of digitalisation beyond industrial contexts is the rise of the Internet of Things (Gubbi et al. 2013) that has started to immerse ourselves in smart environments such as smart cars and smart homes.

Whether the literature describes autonomous control attributed to smart resources such as *holonic manufacturing systems* (Van Leeuwen & Norrie 1997), smart commodities as in *intelligent products* (Meyer et al. 2009) and *digital representatives* (Hribernik, Warden, Thoben & Herzog 2010), or autonomous logistic entities (Schuldt 2011, p. 43): their respective traits align so well with the defining characteristics of software agents, that those have been described as a natural design choice for the implementation of aforementioned concepts (Schuldt 2011, p. 65). This argument goes back Jennings (2001, p. 35) who already stated in 2001 that the agent-oriented software engineering paradigm is well-suited for the implementation of complex and distributed systems.

Access to adequate knowledge is an essential prerequisite for an effective delegation

of process control to intelligent agents that act on behalf of original process stakeholders. Knowledge affords the agent with informed perception, deliberation, and means-ends reasoning (Langer et al. 2006, p. 277).

It is however tedious or even infeasible to try and anticipate and consequently engineer an adequate knowledge base at agent design time. This is due to the dynamic environments in which agents are to act autonomously. The agents themselves therefore have been equipped with the ability to actively manage their own knowledge (Langer et al. 2006, p. 281). Paradigmatically, Gehrke (2011) introduced relevance-driven knowledge acquisition, i.e., the situational acquisition of information, for intelligent agents driven by the desire to render informed decisions throughout their life cycle. This contribution focussed on the systematic retrieval of pieces of information that were previously missing in the agent knowledge base.

Section 2.1 recaps the essential characteristics of intelligent agents. It consequently proceeds with the introduction of common architectural designs for intelligent agents with a focus on knowledge as an essential focal point of agent development.

Beyond information and its context, the knowledge base of an intelligent agent can also comprise complex functions that support situation awareness, decision making, or forecasts about the effect of exerted actions. When intelligent agents are designed to not only exploit such functions in their primary tasks, but to compile and subsequently operationalise these functions based on the information in their knowledge base, thus shaping their knowledge and by consequence behaviour in response to the environment, these learning agents fulfil an essential contribution to knowledge-oriented flexibility in autonomous control.

Section 2.3 consequently extends the previous concept of intelligent agents with learning aspects. These include the architecture of learning agents and the integration of learning and primary agent tasks.

While the implementation of learning capabilities in the design of intelligent agents can pose a challenge in its own right, Section 1.2 identified an even greater challenge that emerges in multiagent communities which are formed by intelligent agents with individual learning capabilities. Namely, what would be a feasible approach to allow a learning agent to benefit from learning activities of its peers.

Section 2.4 argues that the standardisation of multiagent systems with respect to infrastructure, message exchange and interaction protocols along with contributions to multiagent organisation, constitutes the foundation for a potential discourse-based approach for the enhancement of individually trained models for decision support, specifically classification models.

The development of multiagent systems that are characterized by a significant amount of emergent discourse-based interaction requires a sound strategy for validation and investigation of system characteristics.

To this end, Section 2.5 explores multiagent-based simulation as an empirical evaluation methodology and introduces a suitable multiagent platform. Finally, the treatment of agent technology is summed up in Section 2.6.

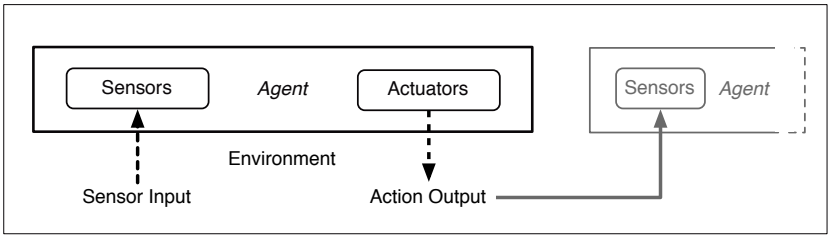



**Figure 2.1:** An agent embedded within its task environment. The agent perceives partially or in full via its sensors. Its actuators can be employed to act upon the environment (adapted from Wooldridge (2002, p. 16)). If the environment hosts additional agents and affords the potential for (direct) interaction, this qualifies as *multiagent environment*.

## 2.1 Intelligent Agents

The properties of a given task environment determine the scope of information that an agent embedded therein can perceive through its sensors. These properties moreover determine the kinds of actions an agent can evoke on the environment and the possible results thereof (Figure 2.1). To accomplish a particular task, an agent has to decide on appropriate actions based on its perception.

This decision making which comprehends deliberation and means-end reasoning results in the exhibited agent behaviour. In mathematical terms, it can be conceived as a function which maps the perceived input to the action output (Russell & Norvig 2010, p. 35). As shown in the subsequent section on abstract agent architectures, the function view immediately reflects simple stimulus-response agents. A functor interpretation however accommodates architectures with an internal agent state.

While the agent concept on an abstract level applies to both biological and artificial agents alike, it is the latter group which is hereafter characterized in more detail. Artificial agents initially fall into one of two groups, depending on the embedding of their respective control program. Robotic agents perceive and act upon the world through a physical articulated body. Agents, on the other hand exist as software programs and reside within a virtual environment.

The design of each artificial agent seeks to serve a particular purpose or design goal as it is the case for each computer program. Typically, that purpose is derived from a strategic long-term goal state that the agent is to reach by following suitable action plans.

Throughout its life cycle, an agent is intended to act in a rational manner, flexibly selecting its activities out of a portfolio of alternatives dependent on the desirability of their expected outcomes from the point of view of the agent (Russell & Norvig 2010, pp. 36–38). This specification is ensured by an external, invariant performance measure that can be used continually from within its behavioural function. Hence, the agent design stipulates a level of meta control, that provides a prioritisation among multiple, potentially conflicting agent goals and the means-end plans that the agent can come up with to effectuate intentions, i.e., currently

endorsed goals. The performance measure hence enables the agent to evaluate its potential activities based on the perception of the task environment and adapt the agent behaviour accordingly.

An agent is considered rational specifically with respect to the preset performance measure, if it attempts to always act in the best available manner to bring about its objectives. Thus, agent rationality is always bound to a concrete design preset whose alignment with general rationality is within the responsibility of the agent designer. Consequently, it is possible yet undesirable that rationality with respect to an invidous performance measure yields an agent behaviour that is considered irrational in common terms.

Consequently, this stresses that one significant challenge in assembling intelligent agents is the development of an adequate agent function that ensures and sustains the desired agent performance. An important aspect with respect to sustainability concerns any activities that can be conducted by the agent in an autonomous manner as to monitor, and ameliorate its performance through adaptation and learning processes. Rationality hence extends to the desire to self-improve continuously.

From an architectural point of view, an agent constitutes an entity within a computational environment that fully encapsulates its behavioural function as well as its internal state and knowledge base. Based on this observation and citing (Craig 2000, p. 3), Berndt identifies similarities with the state and functionality of objects in object-oriented programming. However, in order to delineate the concept of an agent from related concepts, its is necessary to go beyond objects. In fact, concrete agents are often implemented in object-oriented languages in terms of complex object graphs that codify the orchestration of dedicated classes for the various agent sub-systems into a consistent whole.

The essential difference, hence, between agents and regular object-oriented entities lies with the pursuit of an agenda. Traditional object-oriented entities typically implement an interface that constitutes a contract to potential clients of the class with respect to functions that can be provided. Calls to these functions can alter the state of an object. Actually, they are the only way to effectuate state transitions on objects. Thus, object-oriented services are passive and only function within the object graph maintained at run time. External stimuli such as input from a user, scheduled timers, or signals from third-party systems are necessary to yield desired behaviour. Software agents, by contrast do not rely on external stimuli for the conduct of their activities. They are endowed with an intrinsic stimulus to continually pursue their design objectives.

Consequently, agents do not expose an interface in object-oriented terms as these counter all autonomy with respect to the internal state, i.e., objects are manipulated directly. While agents can also expose services to peers and the outside world, it is important to note the different character of service requests. Effectively, all request are filtered in the light of the agent's own agenda.

Hence, while an agent can be requested to perform a certain action, there is no guarantee that the agent actually decides to oblige. It will answer the request iff it is in accordance with its own best interest. In the process, the agent could decide to negotiate terms according to some protocol. It can deliberate on the character of its relation with the caller along dimensions such as trust or status.

Most researchers agree that the autonomy that is implied here constitutes a central pillar of agency (Wooldridge, 1999, p. 28).

**Weak Notion of Agency** With the objective to consolidate various perspectives on intelligent artificial agents and their preeminent characteristics brought forward in the agent research community and beyond, Wooldridge and Jennings have suggested a weak notion of agency. This notion is to convey a minimal catalogue of criteria for an intelligent agent (Wooldridge & Jennings 1995). Four properties are enumerated which an intelligent agent must exhibit.

a) Autonomy

b) Reactivity

c) Pro-activeness

d) Social ability

The mentioning of *autonomy* as the first entry in the preceding enumeration implies its accentuated stance. An autonomous agent possesses both the capability and freedom to decide upon its own behaviour and interaction with its task environment depending on its perception of the environment and its internal state. An agent thus maintains its own thread of activity which can be conceptually understood as a self-sustaining cycle of perception, deliberation, and action. This ability to actively pursue its given objectives without the necessity for external stimuli distinguish an agent from a passive entity.

Although autonomy is a necessary prerequisite for intelligent behaviour, Schuldt remarks that it alone is not sufficient for intelligent agent behaviour (Schuldt 2011, p. 76). An additional criterion is the potential for flexible behaviour. An intelligent agent must be alert to changes in its task environment which bear relevance to the pursuit of its objectives in order to render appropriate decisions and select a feasible course of action. If the agent comes to identify a situation which demands action so as not to deviate from the pursuit of the objective, the agent must engage in activity to handle the situation appropriately. In seeking to do so, the agent adheres to the *reactivity* criterion.

While reactivity refers to the ability to handle disruptions, unexpected events, or in general critical events by means of re-planning and behaviour adaptation, the complementary *pro-activeness* criterion refers to an agent's purposeful action towards its objectives. That is, in exhibiting pro-active behaviour, an agent not solely responds to perceived environmental stimuli but actively takes goal-oriented measures. The latter often implies that the agent relies on some model of its environment and the capability to plan sequences of actions, potentially even accommodating future contingencies. While autonomy implies that an agent is endowed with the freedom to determine its behaviour on its own, pro-activity is necessary to identify appropriate action in a particular situation.

With specific reference to the topic of this thesis, pro-active behaviour may also accommodate objectives focussed on an agent's knowledge including models of environment aspects employed in decision making. Examples include, amongst others, knowledge acquisition, model construction, and model maintenance.

Autonomy, reactivity and pro-activeness together enable an intelligent agent to pursue its design objectives in a flexible and deliberate way. These properties ensure an agent's capability to navigate its environment in an intelligent manner.

To understand the fourth criterion in Wooldridge and Jenning's weak notion of agency, it is adjuvant to note with reference to the literature (e.g., (Abecker et al. 2003, Dignum 2004, van Diggelen et al. 2005)) that in a large number of scenarios, a distinctive characteristic of the task environment hosting an agent is the existence of further agents such that the pursuit of the respective design objectives affords interaction, either mediated via the environment or immediately through communication (see Figure 2.1). In such a multiagent environment, the outcome of agent actions is influenced by the activities of other agents. This situation bears the potential for conflict among agents whose goals are contradictory or which compete for limited resources. It is also possible that the agents try and perform incompatible actions (Timm 2003, pp. 6-10). However, the existence of other agents may as well create novel opportunities for the individual agent as potential peers may possess complementary problem solving abilities (Xing et al. 2009, Schuldt 2011, Bloos et al. 2011).

Hence, cooperation with such agents widens the scope or increases the efficiency of operations. The criterion of *social ability* consequently refers to the capability of an agent to interact with fellow agents in a multiagent setting. Constituent capabilities comprise the identification of suitable interaction partners, the conduct of adequate protocols of interaction, and ultimately the coordination of distributed strands of activity into goal-oriented joint action to help pursue an individual or common goal.

Consequently, social ability extends the scope of agent capabilities from individually rational and flexible behaviour to the social dimension. In this context, it is important to bear in mind, that it lies within the autonomy of the agents themselves to share their respective competencies as services in a particular context. That is, rather than burdening an agent designer with the necessity to anticipate and then explicitly define all concerted multiagent activity at design time, the establishment of cooperation is handled by the cooperation stakeholders themselves.

Since its publication in 1995, Wooldridge and Jenning's weak notion of agency has come to be widely accepted as an agreed-upon point of reference in the agent literature. Stronger definitions exist which are particularly used by researchers in Artificial Intelligence. The strong notion of agency requires characterising agents by mentalistic notions (Wooldridge & Jennings 1995, pp. 118-129).

## 2.2 Agent Architectures

The preceding section identified essential criteria that serve to characterise intelligent agents. Naturally, such a characterisation contains only coarse specifications that can be implemented based on a range of architectural styles. As agent technology matured, agent researchers turned tested agent architectures into patterns similar to design patterns in object-oriented languages. These different

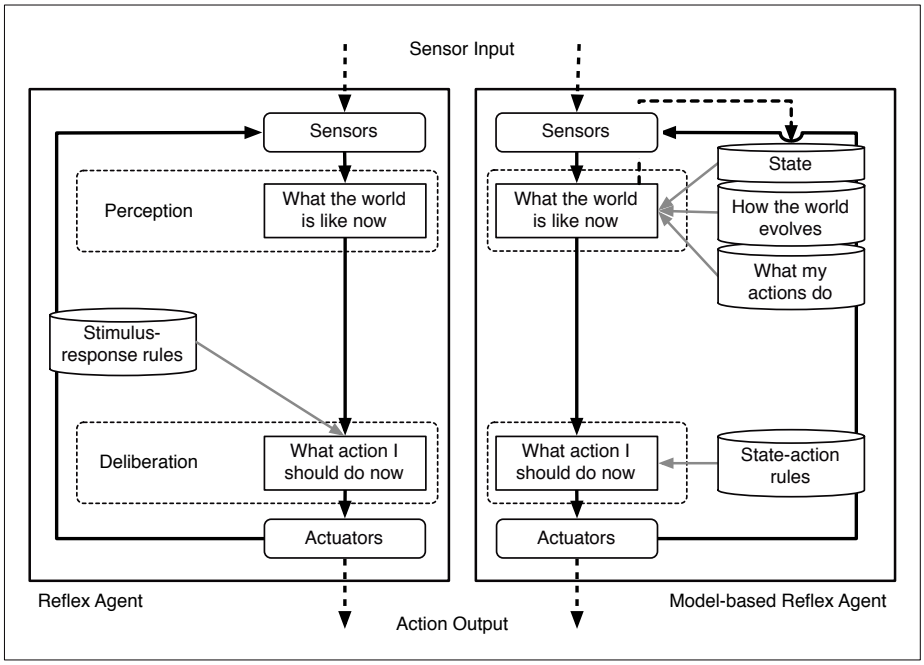



**Figure 2.2:** Abstract architecture for simple reflex agents, adapted from Russell & Norvig (2010, p. 49 & p. 51)

agent types are now introduced with a perspective biased towards the respective use and composition of the agent knowledge base.

The most basic distinction hence is that between agents with and without internal state (Wooldridge & Jennings 1999). This distinction is consequently reflected in the categorisation proposed by (Russell & Norvig 2010) that is reproduced here with additional explanatory elements in Figure 2.2 and Figure 2.3.

The authors distinguish two types of simple reflex agents, that constitute the lower bound for what might be acknowledged as fully-fledged software agent, as well as more sophisticated agents, namely the goal-based agent and the utility-based agent.

The simple reflex agent is the most basic agent architecture introduced by Russell and Norvig. It is shown on the left-hand side of Figure 2.2. The figure has been adapted from the original version so as to incorporate the basic percept-reason-act cycle of a continuous activity that is central in agency. The reflex agent builds on the assumption that a sufficient perception of the environment given the available sensors can be computed from scratch at the begin of each action cycle. Hence, the environment needs to be fully-observable to allow for this style of perception. With regard to deliberation, the reflex agent also assumes a simplistic solution. Namely, that environment states identifiable via perception can be enumerated and associated with a suitable immediate response action. These stimulus-response rules used as a lookup-table constitute the complete agent knowledge base.

To overcome the limitations of the simple reflex agent with respect to perception in partially-observable environments, as shown on the righthand side of

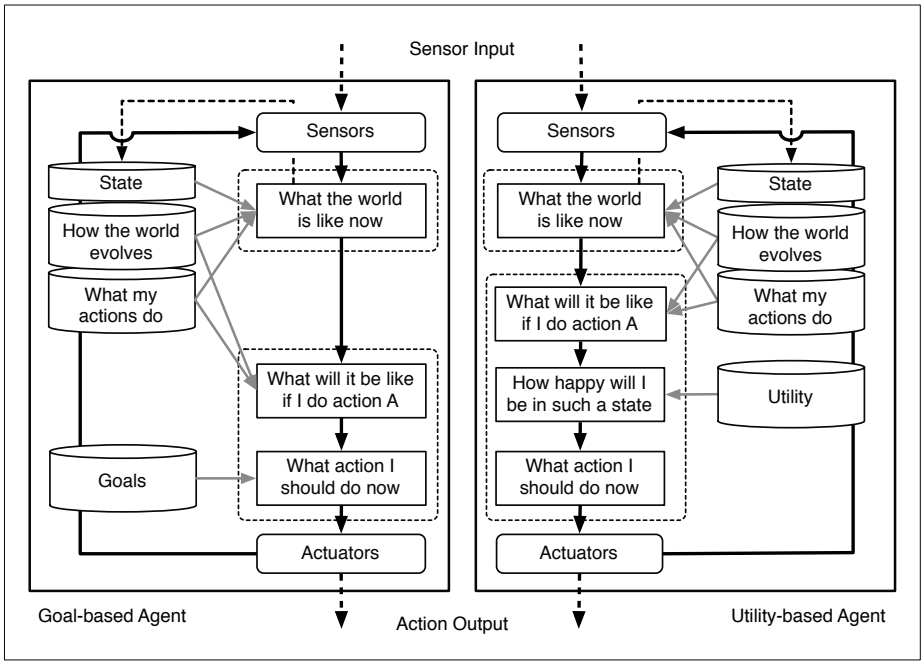



**Figure 2.3:** *Left*: Abstract architecture for a goal-based agent. *Right*: Abstract architecture for a utility-based agent, both adapted from Russell & Norvig (2010, p. 52 & p.54).

Figure 2.2, Russell and Norvig suggest augmented, knowledge-based perception that works incrementally. The agent is equipped with a world model that allows memorising the state of the world at a given point in time. Additional background knowledge pertains to the way the environment is due to evolve over time and what kind of changes are to be expected due to actions executed by the agent. Consequently, given a recent environment state, background knowledge on environment mechanics, and finally updated if incomplete percepts, the agent is enabled to approximate the true state of the environment. Thus, it computes a solid foundation for a continued application of deliberation.

While the model-based reflex agent introduced the idea of a sophisticated perception sub-system, the behaviour specification itself and hence the core agent function still necessitates a comprehensive lookup table of state-action pairs. Specifically, the emphasis is on fine-grained short-term actions as immediate reactions to conditions in the environment. Composing complex strategic behaviour on such a basis is hard to design, evolve, and maintain. Hence, the goal-based agent in Figure 2.3 codifies a desired target state of the environment in terms of a set of discrete goals. Deliberation reasoning is extended such that the agent may revise existing goal commitments given the perceived world state. Adopting a new goal for pursuit necessitates that means-ends reasoning can find a suitable action sequence that would allow for a state transition sequence into the goal state. A prominent example for goal-based agents is the BDI architecture by Bratman (1987) which distinguishes beliefs, desires, and intentions of agents. Rao & Georgeff (1997) propose a modal logics formalism that allows for representing alternative possible world states in the BDI architecture. Conflict management

for discrete goals has, for instance, been addressed with discourse agents (Timm 2003, pp. 86-96).

The utility-based agent shown on the righthand side of Figure 2.3 can be understood as an extension to the goal-based agent. While the latter can only enumerate desirable aspects of a world state in terms of goals, the utility-based agent is able to determine a utility of world states. Hence, issues that root in conflicting goals or actions can be resolved due to prioritisation enabled by a quantitative utility measure. Gehrke (2011, pp. 84–94) provides an overview on probabilistic methods that can be applied to implement utility-based agents.

Concluding the above brief discussion on general agent architectures, it is eminent that as the architectures grow more complex with respect to perception and deliberation sub-systems, the knowledge base that enables these components grows as well. As part of such a knowledge base, it is possible to distinguish state, i.e., the agent's interpretation[1] of the current state of the environment. Further knowledge contributions comprise prediction models that can serve several purposes. To begin with, the paragraph on the model-based reflex agent stressed the use of prediction models to make up for missing observations on parts of the environment. Failure to observe can have two reasons determined by the capabilities of an agent's sensors. First, state variables can be temporarily obscured as in a limited field of view or respective observation can be delayed from that of inducing stated variables. Second, state variables can be completely unobservable. In both cases, prediction models serve to estimate the state of these dependent unobservables based on observable variables. Hence, precise classification or regression estimators are key for a good agent performance.

## 2.3 Learning Agents

As highlighted in the abstract architecture for a learning agent shown in Figure 2.4, the learning module within an agent is but one in a variety of many different interacting agent sub-systems, e.g., contributing to perception, deliberation, and the conduct of actions effecting the environment. *Embodied learning* by an intelligent agent or robotic system therefore draws upon and is to support a wide range of those agent sub-systems introduced in the preceding section.

The circumstance is illustrated in the deliberate dissociation of *learning element* and *performance element* in Figure 2.4 where the latter comprises primary activities. These depend on the specific domain and the task environment where intelligent agents and robotic systems are deployed. Examples mentioned in Section 2 include amongst others both agent-centred domains such as electronic markets (He & Leung 2002), individual transportation (Xing et al. 2009), or autonomous logistics (Bloos et al. 2011, Warden et al. 2012), as well as more robot-centric physical domains like robotic soccer (Warden & Visser 2012).

The learning element hosts the agent facilities to learn and adapt. Although these agent aspects are distinct, they nevertheless interact. On the one hand, learning by an agent is based on experience acquired through continuous interaction

1 also referred to a *beliefs* in the BDI nomenclature

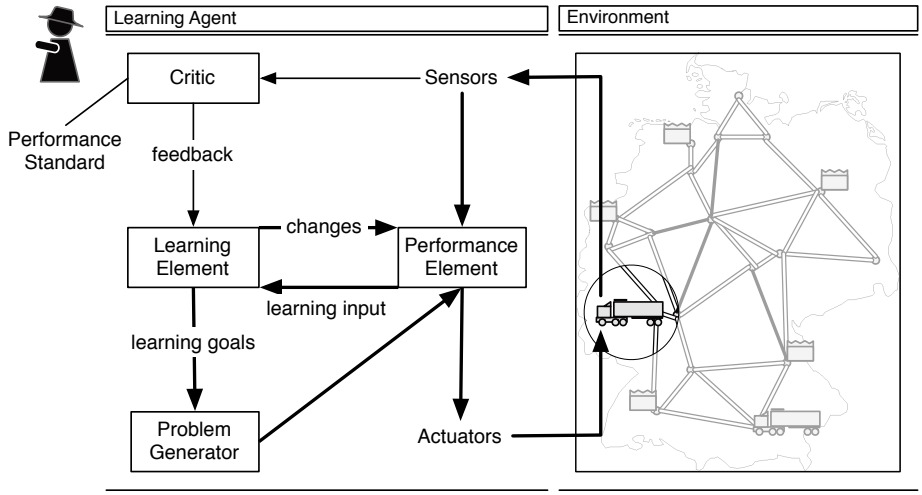



**Figure 2.4:** Abstract architecture for a learning agent, adapted from Russell & Norvig (2010, p. 55). The original design for a non-agent learning system reaches back to Buchanan et al. (1978, p. 35).

with the task environment. On the other hand, the learning element contributes models that enable competitive operation by the performance element.

Besides these central architectural components of learning agents, Figure 2.4 also introduces two additional components with high relevance for adaptive agents. These are denoted as *critic* and *problem generator*.

The *critic* is the component that assesses the performance of the agent or, specifically, particular decision (support) models, with respect to an invariant performance standard. For instance, in the case of a prediction model, such a standard could be an aspired precision and recall. For decision models the performance standard could be directly derived from service specifications. The situation assessment of the critic, which is based on the observation of the environment, is, besides eventual preset strategies, the basis for the scheduling of learning cycles.

The *problem generator* is responsible to suggest to the performance element such actions or courses of action that will lead to new and informative experiences. Hence, it constitutes a means by which the performance element is biased such that it also gives room to the exploration of the task environment, rather than only exploit existing background knowledge and models.

### 2.3.1 Integration of Learning and Primary Agent Tasks

The character of the interplay between an agent's performance and learning element is defined in the following. Kazakov & Kudenko (2001, pp. 255) present different strategies in order to integrate learning with what they denote as recall; the primary agent behaviour wherein learned models are exploited.

Whenever an agent is equipped with learning capabilities is presented with new sensory input through perception, it basically needs to choose whether to simply

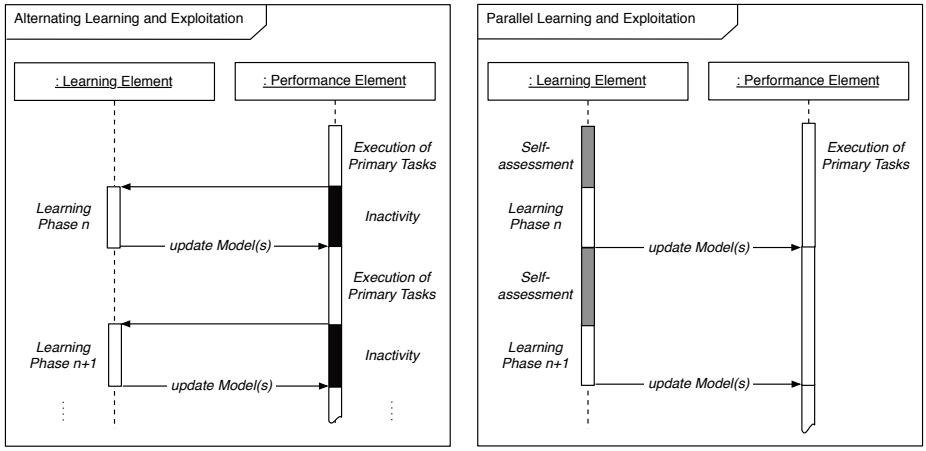



**Figure 2.5:** Modes of interplay between primary agent behaviour and learning subsystem.

register the new input but to defer triggering a new learning phase in favour of drawing from its existing operationalised models of the world for deliberation and means-end-reasoning. The alternative, of course, is to try and update its models immediately to accommodate the new input. Hence, the learning agent needs a reflective meta-reasoning system capable of model-based situation assessment, which controls the scheduling of learning activities.

In the case of *lazy learning* such as *case-based reasoning*, the aforementioned distinction between learning and recall as separate processes vanishes as new theories with a local coverage are created on an on-demand basis during deliberation. If the agent however employs *eager learning* schemes, the authors summarise the the different ways of combining learning and recall as follows:

First, learning and recall could be treated as separate modes with only one of them active at any given point in time. Effectively, the agent then schedules learning phases intertwined with exploitation phases. Second, the agent may conduct learning and exploitation in parallel. In this case, learning is treated as an activity which is orthogonal to the normal agent behaviour. As shown in Figure 2.5, both strands of activity are designed to have synchronised access to the knowledge base of the agent which comprises the acquired data serving as learning input but also any operationalised model in active use by the agent.

In an architecture which allows for parallel learning and exploitation, both the performance element of the agent and the learning element have their own threads of activity with a coordinating meta-reasoner that incorporates the critic and triggers learning.

Given that the model construction in eager learning can be a computationally expensive and time-consuming operation, the design of the critic therefore determines the computation resources that the agent exerts for its learning activities. Particular cases with regard to the circumstances of actual agent deployment may thereby demand custom-tailored answers to the scheduling of learning phases.

For example, consider two functionally identical transport management agents. Let the first be deployed on an agent platform hosted by a high-capacity server

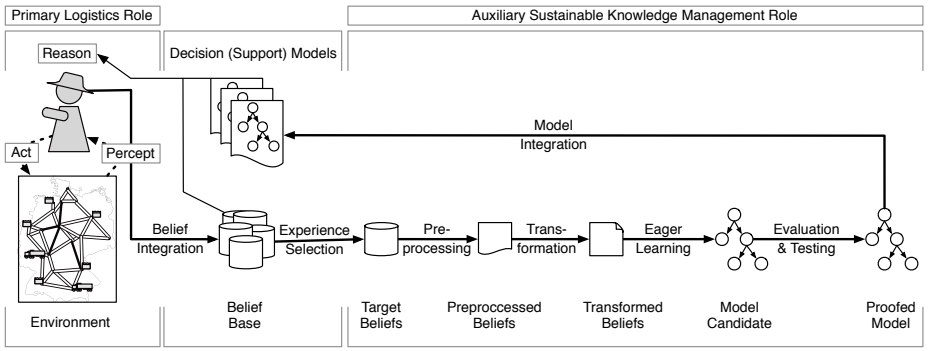



**Figure 2.6:** The steps of an eager learning sweep initiated by the agent in order to construct or adapt a model (Schema adapted from (Fayyad et al. 1996)).

cluster, while the second is deployed on an embedded telematics system attached to its managed truck. In this setting, the latter agent needs to economise with its resources while the former needs not to fear resource bounds.

The concurrent execution of learning and primary activities with only a loose coupling between the two modes acknowledges the potentially high computational costs of the application of machine learning schemes (Warden et al. 2011). While with a step-wise interleave, the execution time for a complete learning phase may have to be bounded in order not to delay deliberation and means-end reasoning, necessitating the use of learning schemes with any-time halting capability or incremental operation, concurrent operation means that the performance element of the agent remains responsive and can always rely on proven models, either pre-crafted or learned and tested autonomously in previous learning phases. Then, whenever the learning element comes up with a replacement model, it can substitute its predecessor in a way that is transparent to the performance element.

### 2.3.2 Application Flow in Agent-based Learning

Figure 2.6 describes the typical steps to be conducted by the learning element of an agent when employing an eager learning approach. That is, the pursued goal of the learning process is the initial construction, or the refinement and/or adaption of an existing model that generalises from particular experience acquired by the agent. The steps in the learning process can be described by analogy to the process for knowledge discovery in large databases (KDD, cf. (Fayyad et al. 1996)), albeit with several notable changes. At the beginning of the learning process stands the belief base into which the agent thus far continuously integrated its observations of relevant excerpts of its situation context (Belief Integration). The agent belief base is the analogue of the data bases considered in KDD. The difference is that all the experience that has been memorised was acquired on the fly during fulfilment of the agent's primary role.

When a new learning phase is scheduled, it is necessary to make a choice which subset of the belief base should be considered for learning (Experience Selection).

This choice has several dimensions. The first dimension is the absolute number of data records that should be used as input to the learner. The strategic options, which are conceivable in guiding the selection, are manifold and bring along characteristic follow-up challenges. For instance, the agent could opt not to make a selection at all for fear of discarding relevant data prematurely. As a consequence, successive learning phases would need to handle an ever-growing data base as input, which is eventually bound to slow down or even inhibit learning late in the agent life cycle. Alternatively, the agent could restrict the data in different ways, amongst others by a restriction based on the absolute number of data records, or the selection of data records from a particular time window in the recent past, or the selection of what is considered a good representative sampling of data records from the belief base. The latter strategies are thereby conceivable in fixed and adaptive form. The actual choice of the selection strategy has an impact on the maintenance of the agent belief base; in particular, for how long experience data should be retained and when it can be consequently safely forgotten since the gist of this experience has been captured in learned models, such that it can still be recollected as needed. A second dimension, which needs to be considered in the selection of experience data, is the weighting of respective data records.

Once the experience selection process is finalised, the next two steps, namely Preprocessing and Transformation, bring the data in a format which is suitable as input for the learning approach that is employed by the agent. In comparison with the KDD case, these process steps have subordinated importance. This is due to the fact that the data gathering and learning entity are one and the same such that problems of data conversion or data processing with regard to missing or incorrect values are much less pronounced.

The next step in the process chain of an eager learning sweep is the application of an appropriate machine learning scheme. The result is a model candidate which might be suitable as a substitute of the respective model that is used thus far in for the execution of the primary agent roles, be it a default model or a self-learned model from a previous learning phase.

In order to assess whether the new model is indeed superior to existing models within the momentary agent deployment context, the next step in the process chain is the model evaluation. To this end, the performance of the model, for instance in terms of precision and recall, needs to be assessed based on benchmark cases that are representative for the respective agent. These benchmark cases can be sampled directly from the agent's own pool of experience. That is, while the eager learning approach can function as a means for the agent to forget aged, and especially repetitive experiences, it must still be ensured that a representative benchmark set is retained.

The final step within a successful learning phase where the model candidate could be affirmed in the evaluation step, is the autonomous operationalisation of the model. As shown in Figure 2.6, this step thereby closes the loop for the current machine learning phase.

It also highlights one major distinction between the process in knowledge discovery in large data bases on the one hand and both active machine learning approaches and general agent-based learning on the other hand. While KDD stresses specifically only the discovery of new knowledge and leaves assessment

and use thereof to the human expert, agent-based leaning, via the critic, puts the learner itself in charge of model assessment.

## 2.4 Multiagent Systems

The agent architectures presented in Section 2.2 give an idea of the heterogeneity of concrete agent implementations in academia and beyond. For those agents acting in isolation within their respective environment, such agent heterogeneity does not pose a problem. However, agents must often cooperate in order the achieve their objectives. Durfee (2001, p. 121) explicates that cooperation among agents is necessary if they cannot solve a task on their own or if they can accomplish their tasks better by means of cooperation.

Cooperation thereby implies that agents within a common environment must posses the ability to notice peers as potential counterparts and have access to effective communication channels. Whether mediated communication via markers in the environment, or direct communication via message transport system, a mutual agreement on both the syntax and semantics of exchanged message content is essential.

While it has been shown that such a mutual understanding can be established dynamically in an evolutionary process (Steels 2003), a more common approach involves standardisation. The scope of standardisation is thereby contingent on the respective multiagent system. For instance, in the RoboCup soccer simulation leagues, technical specifications for inter-agent messaging were traditionally mandated by the league's technical board for each competition. Also, the simulation server ensured that agents could witness the actions of their peers within physical boundaries and provided the message transport system. An additional level of standardisation was then established by the developers of the agent teams.

While in a well-defined scenario such as robotic soccer, communication occurs among agents that belong to a single stakeholder, this is not necessarily the case for agents that are designed to operate in persistent, open multiagent systems as they have been envisioned, for instance, for the implementation of autonomous logistic processes. As Singh (2003, p. 37) notes, interaction between agents from multiple independent stakeholders requires comprehensive interoperability standards. These need to provide appropriate abstractions from different agent implementations, different multiagent platforms, and different vendors (Hellenschmidt & Wichert 2007, p. 97).

The IEEE Foundation for Intelligent Physical Agents (FIPA) (Dale & Mamdani 2001, pp. 3–5) has specified standards aimed at enabling agent interoperability and interaction (Dale & Mamdani 2001, pp. 5–10). These standards comprise 1) an *abstract architecture* for multiagent platforms that involves the facilities necessary for agent life-cycle management, message transport, and service discovery (Foundation for Intelligent Physical Agents 2002a), 2) a *message format* for agent messages, and 3) agent *interaction protocols*.

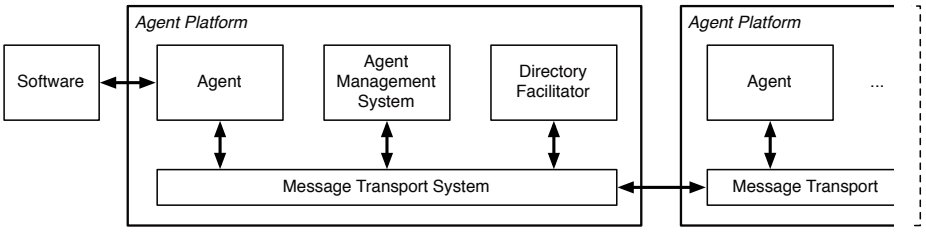



**Figure 2.7:** The agent management reference model as standardised by the Foundation for Intelligent Physical Agents (adapted from Foundation for Intelligent Physical Agents 2004, p. 2)

### 2.4.1 Multiagent Infrastructure

The FIPA abstract architecture standardises essential constituents for fully-fledged multiagent platforms that enable interoperability among multiple heterogeneous agents deployed within a single platform as well as across different multiagent platform deployments and implementations (FIPA, Standard No. SC00001L, 2002). The standardisation mandates that compliant multiagent platforms such as FIPA-OS (Poslad et al. 2000) or the Java Agent Development Environment (JADE) (Bellifemine et al. 2005) comprise the architectural components shown in Figure 2.7.

Probably the most fundamental component is the *message transport service* which provides transparent communication channels among agents deployed within an interconnected cluster of agent platform deployments. The message transport system introduces the necessary abstractions to allow for both agent and agent platform heterogeneity while ensuring effective, unhindered interaction. This entails that individual agents are assigned a globally unique agent identifier (AID).

A single *agent management system* (AMS) is mandatory for an agent platform. It exerts supervisory control over access to and use of an agent platform. The AMS handles agent life-cycle management within the platform. Acting as white page service, each agent must register with the AMS to be assigned its AID. The AMS hence acts as the central agent registry. A yellow pages service should be enabled by a *directory facilitator* component. Agents can register their services with this component and query it for services provided by other agents. Both agent management system and directory facilitator may be implemented in terms of agents[2]. This allows for interacting with them via the message transport system like with any other agent (O'Brien & Nicol 1998, p. 55).

2 These infrastructure agents implement essential platform services. Hence, the requirement for dependability in service provision necessitates a constricted autonomy. Therefore, interestingly, these agents are paradigmatic examples for the fluent transition from classic object-oriented services to the agent paradigm

### 2.4.2 Agent Interaction Protocols

Standardisation of messages exchanged among agents enables mutual understanding in communication over the standardised message transport mechanisms offered by the multiagent platform. A convention on the general message structure with respect to descriptive meta-information such as performative, sender, receivers and applied content language is a sufficient prerequisite for the exchange of single messages.

Beyond this, FIPA acknowledged that effective and efficient conversations require additional means to discern the respective message context. Usually, single messages constitute specific steps in a thread of conversation that follows a mutually agreed-upon interaction protocol. Such protocols may involve multiple stakeholders in distinct conversational roles, most notably initiator and participants. The conversation threads follow the asynchronous nature of textual messaging rather than the synchronous nature of direct speech. Multiple conversation threads states might need to be maintained in parallel across different interaction protocols. Humans engaging in interaction with peers benefit from a culturally induced mutual agreement to everyday interaction protocols and handle interlaced conversations with relative ease, seamlessly switching conversation contexts in the process.

The standardisation process by FIPA accomplishes the same for software agents hence entailed the identification of an essential set of interaction protocols whose application would cover a wide range of conceivable conversations, followed by a formalisation, and, finally, the extension of the ACL message format. Beginning with the latter, ACL allows for the specification of an *in-reply-to* attribute that allows to mark a message as a direct successor of a previous incoming message. The *reply-by* allows for demanding an upper bound to asynchronicity in a message response. Finally, ACL also envisages explicit unique identifiers for interaction protocol instances, i.e., conversations, and their direct reference in constituent messages. Thus, agents are relieved from message-context association and can concentrate on the protocol flow.

FIPA fully specified nine distinct interaction protocols (Poslad 2007, p. 12) as standards.

- synchronous (FIPA, Standard No. SC00026H, 2002) and asynchronous (FIPA, Standard No. SC00028H, 2002) *request* protocols,
- a *query* protocol (FIPA, Standard No. SC00027H, 2002),
- a *propose* protocol (FIPA, Standard No. SC00036H, 2002),
- iterated (FIPA, Standard No. SC00030H, 2002) and non-iterated (FIPA, Standard No. SC00029H, 2002) *contract net* protocols (see Figure 2.8),
- a *recruiting* protocol (FIPA, Standard No. SC00034H, 2002),
- a *propose* protocol (FIPA, Standard No. SC00036H, 2002),
- a *subscription* protocol (FIPA, Standard No. SC00035H, 2002), and finally
- a *brokering* protocol (FIPA, Standard No. SC00033H, 2002).

These protocols cover simple interactions whose purpose falls in one of two categories, namely task assignments and information exchange. On the task side, requests for action, the proposal for action, recruitment and negotiations

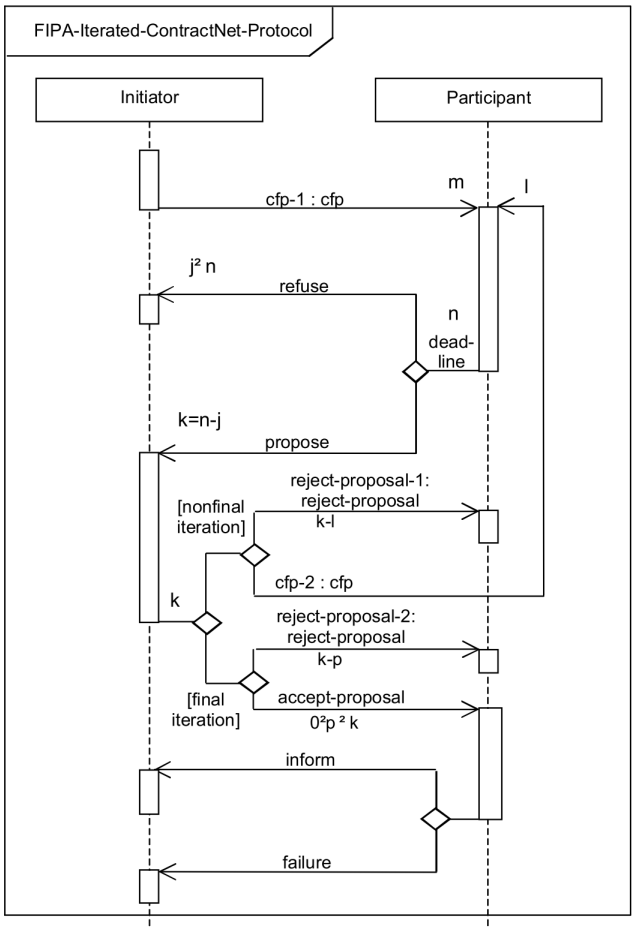



**Figure 2.8:** The non-iterated version of the FIPA contract net protocol, adapted from (FIPA, Standard No. SC00029H, 2002)

for tasks is supported. With respect to information exchange, direct queries, brokerage, and subscriptions are supported. Two additional protocols, that were not officially standardised but are nevertheless frequently used are the *english auction* (FIPA, Document No. XC00031F, 2001) and the *dutch auction* (FIPA, Document No. XC00032F, 2001) protocols.

These agent interaction protocols can be represented in the Agent Unified Modeling Language (Huget 2004, pp. 64–66), which is an agent-specific extension of UML 1.0[3] (Booch et al. 2005). AUML extends standard UML interaction diagrams to cope also with particular requirements of agents, e.g., regarding autonomy (Huget & Odell 2004, p. 16).

Agents participating in interaction protocols are represented by their so-called lifeline (Huget & Odell 2004, pp. 23–24). It consists of a rectangle stating the name or class of the agent and a vertical dashed line below. Messages are represented by arrows between the lifelines of two agents. Messages are generally exchanged asynchronously because it is impossible to directly invoke methods on agents. This is reflected by the open head of message arrows (Huget & Odell 2004,

3 the current major version of UML in the early 2000s, when FIPA finalised its protocol standards

pp. 25–26). Cardinalities can be applied in order to indicate that messages have multiple senders or receivers. Messages are generally labelled by their respective performatives.

More recently, researchers have extended UML 2.0 to more precisely capture the FIPA interaction protocols compared to AUML which has been criticised being rather illustrative in character (Haugen & Runde 2009).

### 2.4.3 Multiagent Organisation

Interaction protocols as introduced in Section 2.4.2, whether defined by FIPA standards or developed for domain-specific purposes, effectively structure the course of agent conversations. Hence, they focus on rather short-term interactions. However, in the same way that single messages may be part of a longer conversations, multiple conversations may constitute steps within a certain thread of activity, aimed at achieving a long-term goal. The respective sustained cooperation potentially requires additional formalisms. These can be manifested through organisational structures. Ferber (2001, p. 114) explains that organisations are characterised by two properties. First, by the respective roles that are assigned, whether from an external party or by the involved agents themselves. Second, by the abstract relationships between these roles. Common generic roles in multiagent systems comprise broker, directory, mediator, and moderator. Broker or middle agents provide agents with services of other agents, e.g., with knowledge acquired by other agents (Langer et al. 2006, p. 282). Directory facilitators (cf. Section 2.4.1) administer lists of agents or services provided. Mediator agents translate between agents that have different languages. Moderators organise negotiations within groups of agents. Apart from these common examples, it is often necessary to define other roles based on the requirements of the concrete application at hand.

Kirn et al. (2006, pp. 46–48) propose to characterise structures in multiagent systems by three dimensions, namely capabilities, duration, and decision-making. The capability attribute refers to the question whether jointly acting agents are homogeneous or heterogeneous in their capabilities. The possible duration of organisations spans from short-term to long-term. The authors distinguish different degrees in freedom for the decision-making of agents. Depending on the specific structure, the autonomy of agents may be restricted, e.g., by democratic decisions or hierarchical orders. In economic systems, an additional fourth dimension helps distinguishing whether interaction is carried out horizontally, vertically, or diagonally. This classification allows, for instance, for distinguishing cartel, collaboration, cooperation, alliance, department, institution, and location (Timm et al. 2006, p. 47).

### 2.4.4 Multiagent Platforms

A number of attempts have been undertaken to compose surveys on the growing number of multiagent platforms which are developed and employed in the agent

research community. For instance, Bordini and colleagues (2006) provide an overview of agent-oriented programming languages and underlying agent platforms. Weiß and Jakob dedicate several chapters in (Weiß & Jakob 2005) to a description of multiagent platforms such as the Zeus-Toolkit, JACK (Winikoff 2005), or JADE (Bellifemine et al. 2007). The authors also present a feature comparison based on evaluation criteria such as platform security, completeness, scalability and adoption. Nikolai & Madey (2009) have compiled taxonomies for a comprehensive list of toolkits for agent-based modelling which focusses on adopted programming languages, license models, platform availability and application domain.

An important criterion for agent-based applications which attempt to go beyond particular academic questions is interoperability. Standards for agent interoperability have been issued by FIPA (cf. Section 2.4.1). Hence, it is desirable that a multiagent platform is chosen right from the beginning whose adoption of the FIPA standards already accommodates the consideration of cross-platform agent communities.

The strict adherence to FIPA standards only holds for some of the systems investigated in Weiß & Jakob (2005), and indeed for only the minority of available multiagent platforms. Bellifemine et al. (2007) claim that of the FIPA-compliant platforms, the Java Agent Development framework (JADE) is the middleware with the broadest adoption, particularly in academia.

JADE has been developed jointly by Telecom Italia and the University of Parma (Weiß & Jakob 2005, p. 202). It is a pure-Java multiagent platform with support for distributed deployments where software agents share a common runtime environment across multiple hosts. Possible JADE deployments can scale from large-scale distributed scenarios down to comparatively resource-bounded platforms. In the early days of JADE, the Lightweight and Extensible Agent Platform (Bergenti & Poggi 2002) (LEAP) pioneered the deployment of JADE agents on mobile devices such as Android mobile phones or tablets. Such developments are still ongoing (Bergenti et al. 2014). They enabled also the use of JADE on embedded platforms such as the Intelligent Container (Jedermann et al. 2006). Using the Web Service Integration Gateway technology (Greenwood & Calisti 2004), JADE agents can interoperate with web services. As investigated in (Hribernik, Warden, Thoben & Herzog 2010), this affords for instance an integration of multiagent-based autonomous logistics (Schuldt 2011) with the Internet of Things (Uckelmann et al. 2011). Finally, JADE has been extended to support the implementation of cognitive agents that adhere to the BDI architecture via the JadeX project (Pokahr et al. 2005).

The architecture of a distributed Jade multiagent platform is described in (Bellifemine et al. 2007, pp. 32–34). As shown in Figure 2.9, a complete agent platform is composed of a set of agent containers. These containers are populated by the software agents. A designated main container is responsible for the coordination of the remaining registered agent containers. In compliance with the FIPA reference model, which has been introduced in Section 2.4.1, the main container hosts dedicated agents that implement the agent management system (white pages service) and the directory facilitator (yellow pages services). By means of the message transport system, JADE agents can exchange messages. The hosting container is thereby responsible for message delivery within the container, across

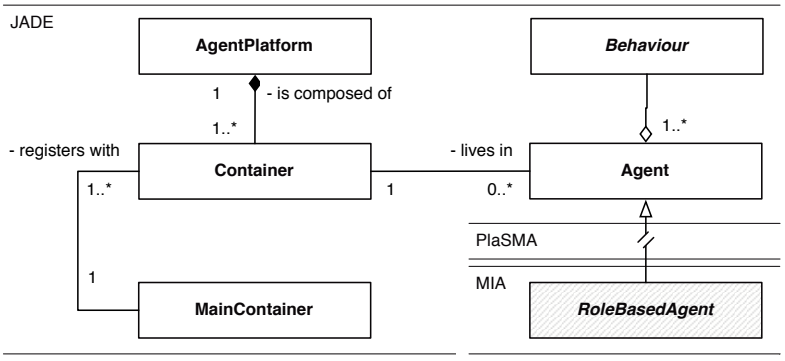



**Figure 2.9:** UML class diagram which shows the relationship between the main architectural elements of a Jade multiagent platform. The software agents that reside on the platform populate agent containers which may be distributed across nodes in a computer network. The runtime behaviour of the agents is modelled by means of behaviour classes. The diagram shows only the root element of the behaviour hierarchy. (adapted from Bellifemine et al. 2007, p. 33)

different registered containers and beyond the borders of the multiagent platform. Software agents act in parallel within their respective container. Each agent is executed as a dedicated operating system thread.

The actual behaviour of the agents is implemented by behaviour classes. The JADE API provides the behaviour hierarchy (Bellifemine et al. 2007, p. 92) depicted in Figure 2.10, that distinguishes simple and complex behaviours. A simple behaviour thereby constitutes an atomic behaviour of an agent with a well-defined, manageable functionality. Behaviours derived from the `SimpleBehaviour` class implement common schemes with regard to timed scheduling and termination. Composite behaviours manage a number of child behaviours and implement a particular scheduling strategy for these children, namely a serial execution of child behaviours such that the active child needs to complete its operation before its successor is started or a pseudo-parallel execution where each active child is scheduled in an interleaved round-robin way. For serial behaviours, there is finally a further distinction which alludes to the possibility for child re-entry. While the sequential behaviour enforces the child scheduling as a fixed sequence, the finite state machine behaviour affords the agent designer the flexibility of the eponymous finite-state automaton.

Due to the possibility of nesting, composite JADE behaviours can be employed to engineer comprehensive behavioural patterns. These can scale to primary domain tasks associated with an agent performance element as well as knowledge management functions associated with its learning element. At the same time, modularity and functional composition of this modelling approach retains the desired maintainability and reusability. A special kind of well-structured behavioural patterns with contingencies constitute conversations among agents that follow interaction protocols, such as the general purpose protocols enumerated in Section 2.4.2 and domain-specific protocols as in the team formation protocols presented in Schuldt (2011, pp. 120–130). For this reason, the FIPA interaction protocols have, for the most part, been modelled in JADE as finite state machine behaviours. This approach allows for the nesting of interaction protocols and

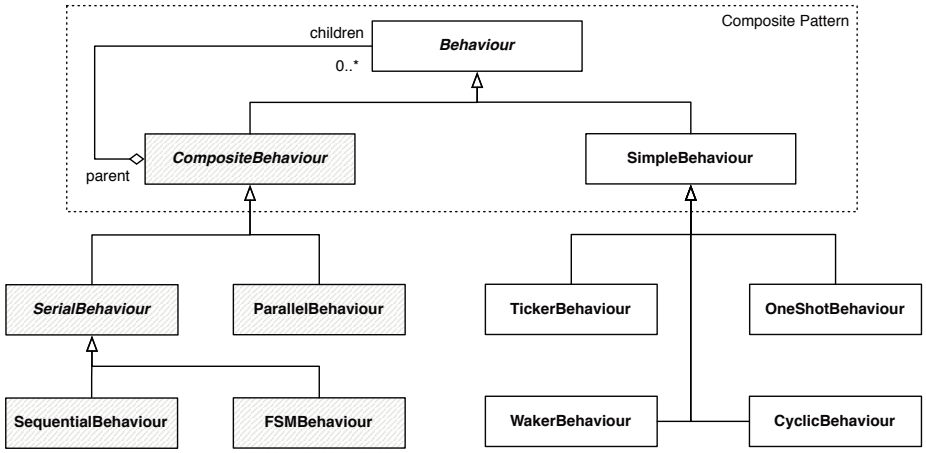



**Figure 2.10:** UML class diagram of the JADE behaviour hierarchy which constitutes the basis for modelling the agent roles presented in Sections 6.3.1 and 6.3.2. (adapted from (Bellifemine et al. 2007, p. 92))

hence complex multi-tier interaction.

Considering the belief-desire-intention school of agent modeling, nested composite behaviours can correspond to concrete action plans with contingency management. Hence, the means-ends reasoning performed by a BDI-agent could be implemented to select intentions, i.e., adopted goals, from a library of stored action plans, so as not to invoke planning from scratch (cf. Section 2.2).

## 2.5 Multiagent-based Simulation

The versatile interaction patterns among agents in multiagent systems entail that an analytical examination of these processes is often impractical. The situation is exacerbated further if not only the interaction, but also the agent behaviours themselves are complex, as it is the case with learning agents. As a consequence, simulation as an evaluation means has traditionally played an important role in the research and development of multiagent systems (Lees et al. 2005).

### 2.5.1 Types and Objectives of Simulation

Herrler & Klügel (2006) pose that the concept of computer simulation is generally understood as the conduct of experiments with a model of a dynamic system. While this model constitutes an abstraction from the original system, it still needs to capture its essential characteristics. In a simulation experiment, the behaviour for the simulation model can be observed over time and resulting output variables, i.e., key performance indicators, can be evaluated to draw conclusions about the original system. Herrler and Klügel distinguish different purposes of simulation studies. Predictive simulation models are constructed in order to assess the behaviour of the modelled system when configured with a specific set of input parameters. Such models are valuable in engineering contexts

with regard to system validation and performance assessments. A second class of simulation models, namely explanation models, is constructed in order to prove hypotheses about the original modelled system or gain insights into hitherto unknown interdependencies of modelled entities or model variables. Davidsson et al. (2007) classify multiagent-based simulation models further into models for prediction, verification, training and analysis.

Simulation techniques can also be classified according to modelling granularity (Davidsson 2001, Herrler & Klügel 2006). The important dichotomy is drawn between macro simulation and micro simulation.

Macro simulation treats the modelled system as a single monolithic entity. Variables on the global level are used to describe its properties. System Dynamics (Forrester 1961) and generally equation-based modelling are paradigmatic techniques in macro simulation.

Micro simulation models, by contrast, involve a potentially large set of active entities whose local behaviours are modelled explicitly and can vary across the population. These entities act within a common environment and influence each other either through their actions or explicit interaction processes. It is from these local behaviours that a global behaviour on the system level emerges.

Object-oriented simulation (Roberts & Dessouky 1998) (OOS), discrete-event simulation (Fishman 2013) (DES), and most recently multiagent-based simulation (MABS) constitute by trend examples of micro simulation. The latter qualification is necessary. Although the modelling of individuals is focussed in micro simulation, the modelling granularity, for instance in MABS, can vary significantly from simplest entities such as single products to global corporations. Schuldt (2011) discusses this issue for autonomous logistics. DES and OOS differ from multiagent-based simulation in that the modelled entities tend to be rather passive entities whose actions over the course of a simulation is determined by a set of transition rules distinct from these entities which is applied by an external simulation driver.

Multiagent-based simulation, by contrast, applies the concept of multiagent systems to simulation (Herrler & Klügel 2006) in that each agent fully encapsulates its behaviour and constitutes a logical simulation process or simulation driver. Borshchev & Filippov (2004) offer a side-by-side comparison of the aforementioned micro simulation techniques. Finally, in a much-cited direct comparison of equation-based modelling and agent-based modelling for simulation purposes, Van Dyke Parunak et al. (1998) conclude that the latter "*is most appropriate for domains characterized by a high degree of localisation and distribution and dominated by discrete decision. Equation-based modelling is most naturally applied to systems that can be modelled centrally, and in which the dynamics are dominated by physical laws rather than information processing.*"

### 2.5.2 Advantages of Multigagent-based Simulation

Davidsson (2001) states that "*if we compare MABS to traditional DES we find that it has several advantages. Just like OOS, it supports structure preserving modelling and implementation of the simulated reality. That is, there is a close*

*match between the entities of the reality, the entities of the model and the entities of the simulation software.*" This statement reflects that multiagent-based simulation originally saw considerable success in the simulation of social, economic and biological systems (Davidsson et al. 2007, p. 22). In fact, in areas such as autonomous logistics and agent-oriented knowledge management, the aforementioned entities in reality, simulation model and simulation software coincide for the most part. Specifically, Gehrke & Schuldt (2009) have proposed the paradigm of *uniform agent design for simulation and operation*. Not only do they acknowledge, similarly to Van Dyke Parunak et al. (1998), that it is natural to evaluate characteristics of artificial multiagent systems in a matching simulation environment. They further promote the idea that it should be even easily possible to migrate a tried and tested multiagent system from the MABS testbed into a productive deployment.

Davidsson (2001) and Herrler & Klügel (2006) catalogue properties which argue for multiagent-based simulation. The central argument is the desired modelling of potentially large numbers of active entities which are spatially or organisationally distributed within their environment. The modelled entities are characterized by pro-active behaviour wherein they follow their own agendas and can act without external stimuli. As they seek to solve problems, or to adapt to their environment with the help of others, and hence dynamically decide to enlist help of others or work as a team, process interdependencies are created which are otherwise difficult to model. Furthermore, individual properties of single agents can be accommodated by the simulation model.

From a technical point of view the MABS paradigm supports distributed computation in a quite natural way. Since each agent is typically implemented as a separate piece of software corresponding to a process (or a thread) it is straightforward to let different agents run on different machines. This allows for better performance and scalability. Finally, Davidsson argues that since each agent is typically implemented as a separate logical process and is able to interact with peers using a common language, it is possible to model fluctuations in the participants of a simulation without interruption (Davidsson 2001). Also, due to the structure-preserving mapping between reality and simulation, it is even conceivable to replace an agent with its corresponding simulated entity such as a real person, for the scope of a simulation experiment. Speaking more broadly, artificial and human actors can participate and even interact within the same simulation. Wurst (2005) presents an evaluation of distributed knowledge management with respect to the performance of mediation methods under different degrees of knowledge heterogeneity, using agent-based simulation.

### 2.5.3 PlaSMA Multiagent Simulation

With the growing popularity of agent-based modelling and simulation in a variety of research fields beyond distributed artificial intelligence, the spectrum of systems for multiagent-based simulation has spiked (Abar et al. 2017). Overviews of respective systems have also been provided by Herrler & Klügel (2006, pp. 581–584), Castle & Crooks (2006, pp. 26–35), or Nikolai & Madey (2009). Many

systems are intended for social simulation. Others focus rather on the evaluation of intelligent agents and their interactions in real-world scenarios.

Due to its predominance among multiagent systems (cf. Section 2.4.4), it is particularly desirable that software agents implemented with JADE can be easily evaluated bz simulation. This issue has been addressed by PlaSMA (Gehrke & Ober-Blöbaum 2007, Warden et al. 2010), which stands for *Platform for Simulations with Multiple Agents*. PlaSMA provides a simulation control middleware for JADE that handles experiment initialisation, time and agent lifecycle management.

A core component of PlaSMA is simulation time management. In distributed simulations, logical processes such as agent threads run concurrently on different platforms or processors within a network, each one characterized by different computational resources. In addition, the logical processes may involve different computationally intensive tasks. Hence, simulation time depends on the CPU and the computational needs of each logical process. Consequently, each process has its own local virtual time. In multiagent-based simulation, logical processes are implemented as agents that usually run as operating system threads. Thus, agents even run pseudo-concurrently on a single CPU simulation platform.

As long as agents are independent of each other, concurrency does not matter. But problems may arise whenever agents interact. Consider an agent passing a message to another agent that is advanced in its local virtual time. The recipient of such a straggler message might have taken other decisions if it were aware of the message on time. This is denoted as the causality problem (Fujimoto 2000). In order to guarantee correct simulations, events have to be processed in accordance with their timestamp order. Diverging local virtual times are addressed by synchronisation which can be either optimistic or conservative. Optimistic synchronisation in principle allows for diverging local virtual times to speed up simulations. It manages causality problems with appropriate roll-back strategies. These rewind an agent operations in time when necessary. Conservative synchronisation, by contrast, strictly maintains a global virtual time across the simulation and hence among all agents that participate in a simulation experiment.

PlaSMA implements a conservative synchronisation strategy. The platform also satisfies additional quality criteria for multiagent-based simulations which have been identified in (Gehrke et al. 2008). Specifically, PlaSMA offers *time model accuracy*, allowing the simulation designer full control over minimal simulation time granularity. It also ensures that message passing between agents within the simulation takes an appropriate amount of simulation time such that, for instance, a message cannot be sent by one agent and received by another within the same simulation time step. Hence, this simulation system transparently ensures *causality* with respect to message passing. Finally, PlaSMA also guarantees *reproducibility* with regard to message reception by imposing an order on messages in an agent inbox.

Simulation control in PlaSMA is jointly managed by two kinds of instances: one top-level controller and a sub-controller for each processor or computer in distributed simulation settings. Each sub-controller locally handles the commitments of its respective agents concerning wake-up timestamps and transmits the minimal commitment to the top-level controller. In return, the top-level controller sends time events to the sub-controllers based on the commitments it received. As noted

before, the internal message handling has been adapted in order to guarantee adequacy, causality, and reproducibility of simulation runs. Time progression and correct message delivery is conducted transparently.

PlaSMA has been extended with a physical simulation world model (Warden et al. 2010). It is comprised of a directed multimodal graph which represents a traffic infrastructure. Physical entities located on this graph are specified in a declarative, formal and explicit model which is expressed as an OWL-DL ontology (Bechhofer et al. 2004). The scenario designer can rely on a repertoire of extensible domain ontologies which cover important concepts with respect to main PlaSMA application domains, that is transport and production logistics as well as urban mobility and public transport (Warden et al. 2010). The scenario ontology models the initial state of physical objects within the simulation environment at simulation start. The initial association of agents as digital representatives of managed objects is established as part of the scenario configuration. PlaSMA provides a comprehensive API which allows simulation agents to manipulate managed physical objects via actions (Warden et al. 2010).

Environmental dynamics can be modelled in PlaSMA through environment agents. These can change the topology or properties of the underlying infrastructure graph, physical objects thereon or even create/destroy physical objects at runtime. More recently, the PlaSMA world model is extended to allow for the direct import of real-world, large-scale traffic infrastructures from the Open Streetmap project (Edelkamp et al. 2013, Greulich et al. 2013), including public transport infrastructures.

PlaSMA also comprises tools for simulation visualisation and the specification of a scenario-specific system of key performance indicators. For these performance indicators, measurements are taken by one or more simulation agents over the course of simulation runs. The measurements are logged persistently into a database and are hence available for evaluation purposes right after the completion of a simulation experiment.

PlaSMA supports the conduct of experiments which feature a number of consecutive simulation runs. These runs are conducted with different seeds for random number generation, based on an initial seed specified in the scenario configuration.

The same configuration also allows for the specification of simulation agents that participate in an experiment with their respective heterogeneous properties. For convenience, this process can be carried out via PlaSMA's visualisation client or via direct manipulation of the scenario XML-file. Finally, PlaSMA features a batch client which allows for executing a number of preset experiments with repeated simulation runs without further intervention by the experimenter.

Started as an academic project, PlaSMA has become a mature simulation platform which has served as a joint simulation platform for the Collaborative Research Centre 637 "*Autonomous Cooperating Logistic Processes - A Paradigm Shift and its Limitations*" (see Gehrke et al. 2010). Although in that capacity its main application focus has been the simulation of autonomous control in transport logistics (Bloos et al. 2011, Warden et al. 2012), PlaSMA has also been applied successfully for public ride sharing (Xing et al. 2009) and recently urban mobility (Edelkamp et al. 2013).

Experiences with the integration of the AQ21 rule learning system (Gehrke &

Wojtusiak 2008*b*) and the Learnable Evolution Model (Wojtusiak et al. 2011, 2012*a*,*b*) have shown that PlaSMA is alsosuitable to conduct learning centred experiments.

## 2.6 Chapter Summary

Multiagent systems composed of intelligent, learning agents have been identified as a viable and often natural choice for the implementation of complex applications that need to integrate multiple stakeholders with different organisational backgrounds, that feature an inherent spatial distribution and a distribution of essential knowledge for process control. Where centralised control is infeasible, the paradigm of autonomous control envisions a far-reaching delegation of control to intelligent agents as digital representatives.

Access to sufficient knowledge is thereby critical. As it is infeasible to endow agents with all necessary knowledge at design time, it is essential instead to design learning agents that possess the capability to interleave primary agent tasks with secondary knowledge-oriented functions. In this way, an agent can use learning to improve its various sub-systems, among them perception, deliberation and means-ends reasoning.

A challenging task is the targeted acquisition of data and information that can be used as learning input. Especially in situations where a learning agent resides within a multiagent environment with peer learners, there is a challenge to allow each learner to benefit from knowledge induced by its peers.

The standardisation of multiagent platforms and agent interaction provide the technological foundation for interoperability.

Advances in multiagent-based simulation have rendered accessible tools for empirical studies on multiagent behaviour in controlled environments.

Chapter 3

# Agent-oriented Knowledge Management

The ability to learn and hence to engage in activities to manage and further its own knowledge has been a focal theme in the overview on intelligent agents and multiagent systems in the preceding chapter. What is more, the scenario of knowledge assets that are distributed in an agent community let to the research hypothesis that this knowledge and the establishment of adequate access to it in the context of individual learning would serve to help the individual to exceed self-contained endeavours. Effectively, these themes are also central to the multidisciplinary field of knowledge management which is subsequently discussed from the perspective of both management sciences and multiagent systems.

Initially, Section 3.1 seeks to define knowledge management and introduces basic concepts.

As this thesis is motivated by the desire to enable the transfer of knowledge among software agents, the section then introduces important notions of knowledge, discusses their implications with regard to knowledge management and presents a classification of constituent processes that drive knowledge management.

Having established important concepts and fundamentals, Section 3.2 gives a short overview on the evolution of knowledge management systems. The remainder of the chapter consequently highlights the intersection of knowledge management and multiagent systems from several points of view.

Since knowledge management is traditionally concerned with what might be informally referred to as knowledge elicitation for humans to enable their efficient conduct in the workplace, Section 3.3 gives an overview of *agent-mediated knowledge management* where agent technology is used to enable, or significantly facilitate information and knowledge acquisition. Hence, software agents assume mediator roles between sources of knowledge (corporate, personal, or public domain) and the knowledge consumer.

Finally, Section 3.4 acknowledges that within multiagent systems, the involved agents themselves also rely on adequate access to information and domain knowledge for informed decision-making and planning. In fact, dimensions such

as breadth, topicality, certainty, or task adequacy of accessible data, information and knowledge are critical if an agent seeks to remain competitive in its task environment or even have a competitive edge. Hence, software agents have become knowledge consumers themselves. This development then naturally leads to *agent-oriented knowledge management*; a form of knowledge management whose functions are both procured and utilised primarily by intelligent agents.

## 3.1 Introduction to Knowledge Management

Knowledge has come to be recognised as a central cornerstone for successful and competitive business operation in todays interconnected fast-paced economy. In the strategic management literature, this has been reflected for quite some time with the emergence of a knowledge-based perspective of the firm (Nonaka & Takeuchi 1995) which builds upon and extends the former resource-based theory initially promoted by Penrose (1995).

### 3.1.1 Objectives of Knowledge Management

Organisations have realised that the quality of their respective products and services depend on the effective leverage of the diverse knowledge which is created, maintained and shared by its members, and more recently, also its customers. Considerable resources are spent in research and development to grow the intellectual capital of organisations. Since knowledge-based resources are typically hard to imitate by competitors in a timely fashion and also socially complex, the knowledge-based view of the firm posits that these knowledge assets are valuable and can secure a sustainable long-term competitive advantage.

It has been pointed out, however, that the plain existence of knowledge assets distributed across an organisation is only a necessary, yet not sufficient prerequisite for improved competitive performance (Alavi & Leidner 2001). Even more important is the ability of the organisation to effectively exploit the existing knowledge on a continuous basis. This has been phrased as the challenge to "*provide the right people with the right information at the right time*"(van Elst et al. 2004, p. 2). Backed by Jünemann who stated that the objective of logistics is to provide the right quantity of the right objects at the right time in the right quality for the right price (Jünemann et al. 1989, p. 18), the ensuing challenge is one of 'information logistics'.

Knowledge management is primarily a management discipline which combines amongst others methods from human resource management, strategic planning, and change management. It refers to identifying and leveraging the collective knowledge in an organisation to help it to compete (Von Krogh 1998). Guizzardi (2006, p. 20) elaborates on the top-level objectives of knowledge management efforts, stating that organisations should seek to establish an environment that encourages and facilitates the creation of knowledge and hence contributes to innovation. This presupposes the development of a knowledge sharing culture

which enables the unimpeded flow of information among individuals, groups, or corporate departments with the same knowledge interests. The term knowledge creation has been defined as "*a process that organizationally amplifies the knowledge created by individuals and crystallizes it as part of the knowledge network of the organization*" (Nonaka & Takeuchi 1995).

Besides targeting the effective exploitation of knowledge assets, knowledge management also fills a critical role in market environments where the workforce is subject to constant change and geographically distributed over multiple operating sites. Under such circumstances knowledge assets need to be kept within organisational boundaries. Specifically, knowledge needs to be actively perpetuated beyond the use by its original creator.

While the concept of capturing and consequently disseminating knowledge within organisations has a long tradition (Alavi & Leidner 2001), it has originally been approached as a process where knowledge has been captured at the management level, processed and finally disseminated in the form of training programs, documents or manuals. Guizzardi notes that knowledge management adds "*the dimension of potentially using enabling information technologies (such as the Internet, Intranets, data warehouses, data filters and software agents) to support the systematic creation, integration, and dissemination of knowledge*"(Guizzardi 2006, p. 20).

### 3.1.2 Notions of Knowledge and Implications for Knowledge Management

The development of a detailed understanding of the scope of knowledge management functions and the respective role of information technology, in particular multiagent technology, presupposes a working definition of knowledge as core subject matter. As several authors interested in knowledge management have pointed out, knowledge is a rather vague concept (Maurer 2003). The quest for a precise definition reaches back to classical Greek philosophy and has led to considerable epistemological debates. Considering views on knowledge as discussed in information technology, strategic management and the organisational theory literature is to uncover the assumptions that underlie organisational knowledge management.

**Hierarchical View of Data, Information, Knowledge** A popular approach to the definition of knowledge in the IT literature goes back to Ackoff (1989) and focusses on the delineation of data, information and knowledge. Ackoff introduced the concept of the knowledge pyramid. It implies that data, defined as basic, discrete, objective facts constitutes the ground layer of this hierarchy. Data is considered to be available in abundance, yet on this elementary level, the pieces of data per se are hardly useful. Hence, the data is elevated to information whereby the data is put in an interpretive context. Knowledge, then, is defined as information that has been culturally understood such that it explains the how and the why about something or provides insight and understanding into something. On top of the knowledge pyramid, Ackoff places wisdom which is knowledge

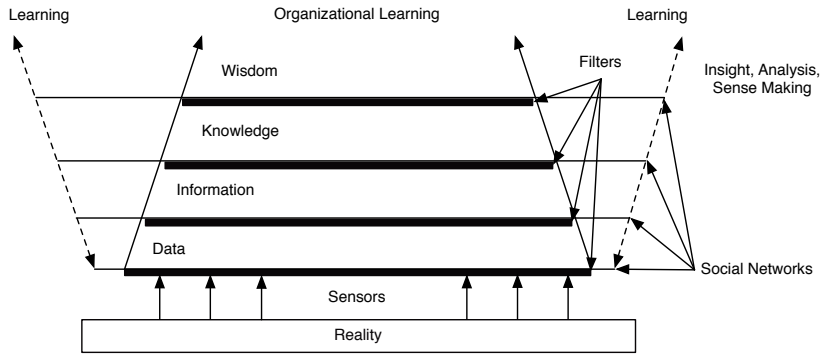



**Figure 3.1:** Revised knowledge pyramid (adapted from Jennex 2009)

which can be applied to different and not necessarily intuitive situations. This hierarchical view has been subject to much discussion and interpretation (see Tuomi 1999, Guizzardi 2006, Jennex 2009). For instance, Tuomi argues that the implied hierarchy from data to knowledge should actually be inverted. Knowledge must exist first before information can be formulated, and before data can be measured to form information. Critical to this argument is the position that knowledge cannot exist outside of an agent. As such, it is fundamentally shaped by its bearer's context of use as well as her initial stock of knowledge. Jennex (2009) combines the positions of Ackoff and Tuomi and introduces a revised knowledge pyramid with special focus on knowledge management activities (see Figure 3.1). Alavi and Leidner follow Tuomi and pose that "*information is converted to knowledge once it is processed in the mind of individuals and knowledge becomes information once it is articulated and presented in the form of text, graphics, words, or other symbolic forms*"(Alavi & Leidner 2001, p. 109). They derive from this view on knowledge that for individuals to arrive at the same understanding of data and information, they must have a shared knowledge background. Besides support for internalisation (information → knowledge) and externalisation (knowledge → information), mediation, i.e., the furthering of a shared understanding, is an important knowledge management objective.

**Pragmatic Characterisations of Knowledge** Besides the hierarchical perspective on knowledge presented thus far, Alavi & Leidner (2001, p. 110) outline several characterisations of knowledge. According to this, knowledge has been characterized as a state or fact of knowing (Schubert et al. 1998). This perspective leads to a focus on enabling individuals to grow their individual knowledge as new relevant information becomes available and subsequently apply it to the best interest of the organisation. A second view is that of knowledge as an object or artefact (Carlsson et al. 1996). This view implies that knowledge can exist in codified form such that it can be stored and manipulated by technical means. Alternatively, knowledge can also be viewed as a process of simultaneously knowing and acting. A more structural perspective is related to the object view, yet emphasises specifically the condition of access to information (McQueen 1998). This view implies that organisational knowledge is to be organised in a form which manages and potentially facilitates access to and retrieval of contents. Finally, again according to Carlsson et al. (1996), knowledge can be viewed as a

capability with the potential to influence future decision making. Watson has extended the capability view in focussing less on the implied capability to invoke specific actions in favour of the capacity to interpret and exploit new information. Specifically, "*learning and experience result in an ability to interpret information and to ascertain what information is necessary in decision making*"(Alavi & Leidner 2001, p. 110).

The presented views and in parts divergent notions of knowledge lead to different perspective views on the objectives and prioritisation of knowledge management. This is also reflected in knowledge management systems (cf. Section 3.2). The prevalent dichotomy can be drawn between the 'product-centric' and the 'process-centric' views. In the former case, knowledge management is developed towards acquisition, tending, and providing access to knowledge stocks. In the latter case, the knowledge management seeks to enable knowledge flow and focusses on creation, sharing and assimilation. This applies both to the organisational landscape as well as along a time axis.

**Tacit and Explicit Knowledge** As an early influential contribution to the theory of organisational knowledge creation, Nonaka et al. (2006) have explicated two dimensions of knowledge in the organisational context which are consistent with the object-mental state perspectives in the previous paragraph: tacit and explicit knowledge (Nonaka 1994). According to Nonaka, the tacit dimension of knowledge comprises both cognitive and technical elements. In this context, the cognitive element refers to the mental models from which an individual draws in decision making. Paradigmatic examples include mental maps, beliefs, paradigms, and viewpoints.

With respect to model-based software agents as artificial decision makers as in Section 2.2 this includes the belief in the agent knowledge base as well as decision support models, for instance for classification or situation assessment.

The technical component of tacit knowledge consists of concrete know-how, crafts and skills, all of which apply to a specific task or situation context. By trend, this technical side also bears a stronger relation to physical action.

The explicit dimension of knowledge refers to knowledge that has been articulated, codified, and lends itself to being communicated either in formal form or natural language. Alavi & Leidner (2001, p. 112) emphasise that tacit and explicit knowledge are not to be conceived as dichotomous states of knowledge. Rather, they are mutually dependent and reinforce one another.

### 3.1.3 Constituent Processes Driving Knowledge Management

With the plurality of notions concerning knowledge, a picture of the scope and character of activities that contribute to an effective management of such knowledge emerges.

According to Davenport & Prusak (2000), most management projects have one of three primary objectives. The first objective can be characterized as knowledge elicitation. Maurer captures the gist of this objective in the saying "*if only our*

*employees knew what our employees know [[. . .]]*"(Maurer 2003, p. 2).

While organisations may accommodate considerable knowledge via their workforce, it takes a systematic assessment effort to locate this knowledge and understand its role in business processes. Such organisational self-consciousness forms the basis to derive surplus value from the identified knowledge assets.

Consequently, a second objective can be brought into focus. Here, the development and fostering of a knowledge-intensive culture is central. Such a culture should encourage and aggregate practices such as knowledge sharing as opposed to hoarding, as well as the proactive seeking and offering of knowledge.

While these first two objectives are addressed in management science, the third objective is somewhat more technical in nature. It refers to the establishment of an appropriate knowledge infrastructure. Knowledge management efforts in this category seek to create knowledge networks among members of an organisation. While with regard to knowledge workers this comprises the procurement of space, time, and incentive to interact and collaborate, it is the deployment of tools, knowledge management systems, that play the role of a catalyst. As highlighted in the following sections, which trace the evolution of such tools for knowledge management, a range of fields in information technology and computer science can provide input to create ever more intelligent, adaptive and proactive solutions. From amongst these fields, notable examples actively embraced in this thesis are distributed artificial intelligence in the form of multiagent systems (Chapter 2), and machine learning (Chapter 4).

Alavi & Leidner (2001) introduce a framework grounded in the sociology of knowledge in order to analyse and discuss the potential roles of information technology in organisational knowledge management. It builds upon the view of organisations as a specific form of social collectives, analogous to multiagent systems, and '*knowledge systems*'. According to the authors, organisations as knowledge systems comprise four sets of socially conducted '*knowledge processes*'.

a) knowledge creation, also referred to as construction,
b) knowledge storage and retrieval,
c) knowledge transfer,
d) knowledge application.

**Knowledge Creation** This set of knowledge processes involves the development of novel content, or revision of existing content within the tacit and explicit knowledge of an organisation. Employing both social and collaborative processes on the one hand and the respective deliberation and learning capacities of the individual on the other hand, knowledge is created, shared, amplified, enlarged, and justified in organisational settings. The creation process builds upon a continued interplay between the tacit and explicit dimensions of knowledge presented in the preceding section. One strives to attain a spiral flow of knowledge wherein it filters through tacit and explicit form and between organisational levels such as the individual or the group.

In a seminal contribution, Nonaka (1994) has identified four modes of knowledge creation. First, *socialisation* refers to the conversion of existing tacit knowledge to new tacit knowledge via social interaction. Second, *externalisation* refers to the

transformation of tacit knowledge into explicit form. Third, *internalisation* refers to the dichotomous process whereby explicit knowledge is used as input to grow tacit knowledge. Finally, *combination* addresses the aggregation/consolidation of (multiple sources of) explicit knowledge, or generally its manipulation.

**Knowledge Storage and Retrieval** The preceding set of knowledge processes, and especially their interplay, caters to the ability of an organization to innovate and learn. However, as Tomasello has pointed out, not with regard to the development of organisations but our species' cultural development (Tomasello 1999), innovation is but one part of the equation for sustained long-term development of knowledge assets. Complimentary processes must be applied as well to retain and perpetuate acquired knowledge so as to allow a cumulative effect (a 'ratchet effect' in Tomasello's terms). Hence, the storage, organisation, and structured retrieval of organisational knowledge, also referred to as organisational memory comprises an important set of knowledge processes. While often focused on explicit knowledge (see Section 3.2.1), tacit knowledge also needs to be accommodated.

**Knowledge Transfer** With regard to the distributed nature of knowledge in organisational contexts, a further set of knowledge processes addresses the challenge of knowledge transfer. Literally, on a mere technical level, one could initially conceive this as the challenge to ensure knowledge is rendered available at those locations where it is needed, and made accessible to eligible parts of the workforce. More importantly, however, knowledge transfer can also be conceived as comprising enabling processes that drive knowledge creation. For instance, communication processes and interaction protocols are important concepts. Gupta & Govindarajan (2000) have proposed a conceptualisation of knowledge transfer with five constituent elements, namely

a) the perceived value attributed to knowledge offered by a knowledge provider,
b) the motivational disposition of the provider with respect to the willingness to share knowledge (and spend resources for its procurement),
c) the existence and richness of transmission channels,
d) the motivational disposition of the consumer to acquire knowledge from providers, and finally
e) the absorption capacity of the consumer which is defined as the ability to actually derive added value from acquired knowledge.

Alavi & Leidner (2001) note that up to the turn of the century, the majority of the literature on knowledge transfer revolved around transfer channels. More recently, researchers invested in agent-mediated knowledge management, such as (Dignum 2004, van Elst et al. 2004) have addressed motivational disposition with topics like trust, responsibility and reward.

**Knowledge Application** A final set of knowledge processes addresses the embedding of knowledge exploitation in business processes. As primary mechanisms to that end, directions, organisational routines, and the institution of manageable self-contained task teams have been proposed.

Rooted in the management sciences, Probst et al. (2006) proposed a related classification of knowledge processes or building blocks as constituents for knowledge management. Development processes, distribution processes, knowledge preservation and utilisation find their direct equivalents in the model by Alavi and Leidner presented earlier. Probst et al. (2006) supplement two additional building blocks. Specifically, identification processes are designed to assess which knowledge is embedded within the organisation and determine knowledge carriers and stakeholders. Acquisition is another process type whereby third-party knowledge from beyond organisational boundaries is pulled and integrated into the organisation. These immediate knowledge processes are framed by complementing management-oriented tasks, namely the definition of knowledge goals and the assessment of the knowledge of an organisation.

An important observation which arises from the description of knowledge processes according to the framework by Alavi and Leidner is their intrinsic interconnection. As a consequence, knowledge management consists of a dynamic and continuously conducted set of processes and practices embedded on the individual, group and corporate level. At any point in time and in various parts of an organisation, its members may be involved in different aspects and processes of knowledge management. Knowledge management is thus not a discrete, independent, and monolithic phenomenon in organisations (Alavi & Leidner 2001).

The following section, which turns the focus to knowledge management systems which have over the years been researched and later been actively deployed, carves out that these findings have been taken on with the paradigm shift to distributed, and specifically agent-based knowledge management systems.

The understanding for the knowledge management challenge now puts a notion from the begin of this section into perspective where knowledge management was tentatively equated with information logistics. While analogies to the basic logistics functions, namely transport, handling, storage, and commission, can be identified, it has been established that knowledge management essentially directly addresses knowledge creation. Hence, knowledge is not merely handled. Knowledge management rather (pro-)actively pursues the strategic objective to further knowledge so as to achieve a sustainable advantage in knowledge-intensive tasks.

## 3.2 Knowledge Management Systems

Knowledge management systems have evolved considerably over the course of the past thirty years (Guizzardi 2006). Originating from systems that were based on central repositories of knowledge assembled and maintained by dedicated knowledge engineers, newer developments embrace distributed system architectures, which seek to grant their users full autonomy with regard to their own knowledge, its furthering, and its dissemination (Bonifacio, Bouquet & Traverso 2002). More recently, a trend can be observed whereby knowledge management systems are conceived as intelligent systems in their own right which leverage the qualities of intelligent agents such as autonomy, reactive and proactive behaviour, and social abilities to better accommodate the knowledge needs of its clients (Dignum

2004).

Analysing this evolution from the perspective of agent-based distributed knowledge management, Guizzardi (2006, pp. 26) distinguishes three phases in the early development of knowledge management systems.

### 3.2.1 Centralised Knowledge Management Systems

The first phase has hence been coined by centralised systems that were typically designed and deployed following a top-down approach. Organisation managers backed by knowledge engineers gathered and imposed a structure on the contents of what would become an organisational memory at system design time. Common technologies to implement such centralised organisational memories are relational databases or, more comprehensively, data warehouses. Decision support systems, or specifically expert systems, are another form of knowledge repository which leverages expert domain knowledge and inference capabilities to support decision makers (Luger 2005). Enterprise knowledge portals integrate knowledge management technologies and expose them in accessible form through web-based interfaces (van Elst et al. 2004). Once deployed in the organisational context, the thus disseminated explicit knowledge was than expected to be adopted and potentially updated by the workforce (Fischer & Otswald 2001).

Centralised systems often suffered limited success due to the missing participation of knowledge workers and problem owners in the development process (Guizzardi 2006). Hence, analogous to the paradigm shift in software development away from traditional development models towards agile approaches, the trend shifted towards evolutionary methods to build knowledge management systems, which heralded the second phase of knowledge management systems development. While the basic knowledge management platform was initially developed and evolved proactively in an on-going fashion (Hahn & Subramani 2000), the user of knowledge management systems came to be recognised as stakeholders in the platform development, participating in the elicitation of requirements that would more closely relate to their daily activities (Guizzardi 2006). While the involvement of knowledge workers in knowledge management systems has been acknowledged as a commendable measure, the second wave of knowledge management systems still retained the focus on externalising, aggregating and in the process de-contextualising the knowledge of the individual.

Bonifacio, Bouquet & Traverso (2002) argue that technological architectures for knowledge management systems can be designed in accordance to two different approaches, each of which presupposes a different understanding of organisational knowledge and cognition. The centralised paradigm seen thus far views organisational cognition as a convergent process that collects peripheral raw knowledge from various contributing sources and codifies it into a central repository (Bonifacio et al. 2004). Information technology is thereby conceived as an enabler for central control, standardisation, high availability and robustness. This coincides with notions of knowledge as an object (Carlsson et al. 1996) or condition of access to information (McQueen 1998). Alavi & Leidner (2001) derive the implications that knowledge management under such prerequisites largely

revolves around building and managing knowledge stocks and the procurement of organised access to and retrieval of content. The second, distributed paradigm introduced in (Bonifacio, Bouquet & Traverso 2002) conceives organisational cognition as a distributed process that balances the autonomous knowledge management activities of individual users and groups, and the coordination needed in order to exchange knowledge across different autonomous entities. Corresponding views see knowledge as the state of knowing and understanding or as a process of applying expertise (Alavi & Leidner 2001). Consequently, knowledge management revolves around furthering individual or group learning and understanding through goal-oriented procurement of information. It also comprises the establishment of knowledge flows and the process of creation, sharing, and distributing knowledge. From this perspective, information technology is viewed as an enabler of distributed control, differentiation, customisation and redundancy.

### 3.2.2 Distributed Knowledge Management Systems

The paradigm shift towards distributed knowledge management marks the third phase of knowledge management system development. Guizzardi (2006) notes that this phase is largely characterized by the recognition that knowledge cannot be separated from the communities that create it, use it, and transform it.

This realisation has sparked or enforced the interest in organisations to support and actively foster so-called *communities of practice* (Fischer & Otswald 2001). Bonifacio and colleagues add that distributed knowledge management as introduced in (Bonifacio, Bouquet & Traverso 2002) proposes that the multiplicity of perspectives within complex organisations should not be viewed as an obstacle to knowledge exploitation, as seen before in original two phases of knowledge management systems, but rather as an opportunity that can foster innovation and creativity (Bonifacio et al. 2004). Paradigmatic examples for peer-to-peer architectures for knowledge management systems have been proposed, for instance, with the KEx system (Bonifacio et al. 2004) or the Help&Learn system (Guizzardi et al. 2004).

## 3.3 Agent-mediated Knowledge Management

With the transition to distributed knowledge management systems, multiagent systems have grown in popularity for system analysis, design and implementation (Dignum 2004). Dignum notes that knowledge management environments can be conceived as distributed systems where different actors, each acting with some autonomy on behalf of a user, and each pursuing their own respective goals, need to interact in order to achieve their goals (Dignum 2006). While the ability to communicate and negotiate with peers is a fundamental requisite in such environments, further characteristics in favour of an agent-based approach apply. For instance, the number and behaviour of participants in knowledge management activities cannot be fixed a priori. Also, the system can be expected

to expand and swap constituent parts over the course of the application life cycle.

van Elst et al. (2004) and colleagues state that the notion of agents can be seen as a natural metaphor for the comprehensive modelling of knowledge management environments. These environments consist of a number of interacting entities including individual knowledge workers, groups of individuals such as project teams or departments, and to a growing degree also technical systems. These constitute a potentially complex organisational structure. As this structure can be accommodated in a multiagent architecture, the modelling approach helps to maintain the integrity of the existing organisational structure and the autonomy of its respective constituents. The autonomy and social ability which is constituent of software agents adhering to the weak notion of agency posited by Wooldridge & Jennings (1995) (Section 2.1) are hence basic means to achieve this structure preserving modelling.

The reactivity and proactivity of agents is also seen as an enabler to provide the actors in a knowledge management environment with the flexibility of operation to cope with the 'wicked nature' (van Elst et al. 2004, p. 3) of knowledge management tasks. According to this, adequate solutions to the handling of such tasks do not follow fixed recipes or best practices. Rather, handling strategies depend on the current system state and context. The complex social skills with which agents can be endowed allows for a dynamic formation of task-specific knowledge networks and plans to solve presented tasks.

With respect to knowledge workers for whom the adoption of knowledge management goals typically does not possess high priority in comparison to primary domain tasks, the proactivity and autonomy of agents is also beneficial. On the one hand, tasks can be delegated to assistant agents. On the other hand, given an accurate user profile, the agents can themselves notify their users about new, relevant knowledge.

From a software technology point of view, agents have been recognised as a way to wrap and consequently integrate legacy information systems and data sources into modern distributed information systems. In addition, agents can be used to provide value-added services through service orchestration. They can also provide facilitation services such as mediation in heterogeneous system infrastructures. These arguments emphasise that multiagent systems are an adequate choice of technology in changing system environments which are subject to frequent reconfiguration and growth in order to reflect the change in business processes. Developing and deploying a knowledge management environment based on multiagent technology allows for the gentle, stepwise introduction of knowledge management services and retains future extensibility.

In order to categorise the various approaches to what the authors refer to as *agent-mediated knowledge management*, van Elst et al. (2004, pp. 6) have proposed a description schema with three dimensions which is to capture the whole life cycle of agent-oriented software development. These dimensions are:

a) the stage in a system development process where agents are employed, i.e., in analysis, conceptual design, or implementation
b) the architecture / topology of the agent system
c) the knowledge management functions and the application context in focus.

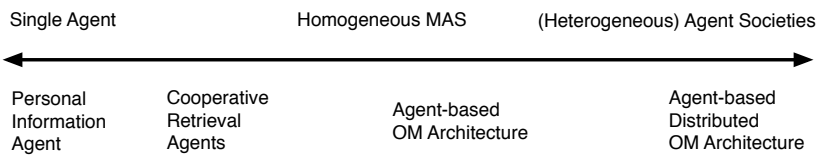



**Figure 3.2:** The *degree of sociability* in agent-based knowledge management systems. Adapted from (van Elst et al. 2004, pp. 8)

**The Agent Notion in System Development** On the system development level, the notion of agents can be beneficial at different stages of the development cycle of an agent-based knowledge management system. The first level comprises an analysis of a given organisational structure as in the identification processes in (Probst et al. 2006) (Section 3.1.3). Often, the entities in the organisation can thereby be mapped in a natural way to an agent representative. This leads to a rather tight coupling between mapped real world and the mapping representation. An organisational analysis is commonly an integral part of methodologies for the development of knowledge management systems such as CommonKADs (Schreiber et al. 1999). Agents are thereby seen as an epistemologically adequate abstraction in order to capture and model relevant participant in knowledge management activities, including their respective roles and tasks, as well as their social interactions. On the level of system architecture, methodologies for agent-oriented software engineering such as Tropos (Bresciani et al. 2004) and Gaia (Zambonelli et al. 2003) can be employed. Finally, on the concrete implementation level, a wealth of agent development frameworks such as JADE (Bellifemine et al. 2007) may be used. While van Elst et al. pose that hence the complete development life cycle for knowledge management systems can benefit from the concept of agents, the authors also caution that for pragmatic reasons, agent modelling may permeate only part of the development process while related technologies like peer-to-peer networks or web services are used for implementation.

**Topology of Agent-based Knowledge Management Systems** Knowledge management systems typically employ a strong organisational top-level perspective on the system of interest such that the macro-level structure is of special interest. Relevant questions on this level ask for the number of agents which provide knowledge management functions. Also, the number of distinct kinds of agents that contribute to an envisioned system and their respective portfolio of roles and responsibilities is an important factor. For systems with a range of agents, the topology with respect to the flow of information/knowledge among the agents, or with respect to their co-ordination is important. Van Elst et al. propose the degree of sociability as a suitable means to characterise the macro-level of agent-based knowledge management systems (cf. Figure 3.2).

*Single agent architectures* constitute one end of the proposed spectrum. Typical examples of such systems include agents which provide the user with a suitable user interface to a knowledge management backend and personal information agents which over time construct a model of their user's interests and behaviour. The latter profile is exploited to filter comprehensive data sources so as to only provide information that is currently relevant to the user. While an effective and intuitive interaction with the human user is paramount for such agents, they still

normally do not employ elaborated interaction with their own kind.

*Homogeneous multiagent architectures* provide a higher degree of sociability in that agents actively co-operate with peers in order to solve their respective knowledge management tasks. Homogeneity here means that such agent systems for the large part consist of only a single type of agent. While their individual goals may differ, for instance when serving different users, their tasks and capabilities are comparable. Agent-based collaborative filtering is seen as a paradigmatic example for this type of system architecture.

Finally, *heterogeneous multiagent architectures* contain a multitude of agent types. Each is thereby typically characterized by different purposes, knowledge and capabilities. Comprehensive agent-based knowledge management approaches such as Knowledge Rovers (Kerschberg 1997) constitute paradigmatic examples of heterogeneous multiagent systems. These systems follow a division of labor approach which favours the employment of various specialists, for instance for the link-up of data sources, integration of various information sources, or presentation of information to the user. These specialists need to cooperate and contribute their respective complimentary expertise in order to procure the knowledge management functions offered by the system.

**Focus of Agent-based Knowledge Management Systems** While the preceding two dimensions for the categorisation of agent-based knowledge management considered system development and architecture, a further dimension is the focus and scope of the knowledge management processes addressed by specific implementations. To some degree, all types of processes which have been identified in Section 3.1.3 can be addressed with agent-based systems.

**Overview of Agent-mediated Approaches** In the following, exemplary applications of agent technology for the development of knowledge management systems are presented. Further comprehensive surveys have been prepared, amongst others, by van Elst et al. (2004) with respect to general agent-mediated knowledge management , and Sato et al. (2012) with focus on agents supporting distributed communities of practice. Additional material can be found in workshop proceedings such as (van Diggelen et al. 2005, Dignum 2004).

Using the topology dimension to structure our account, thereby following the practice in (van Elst et al. 2004), we can first consider single-agent approaches which typically address personal knowledge management and hence concentrate on supporting a specific user in her knowledge-intensive tasks.

**The Single User Perspective: Personal Information Agents** Respective agents act as an intelligent interface between the user on the one hand and data stocks on the other hand. *User interface agents* work on the presentation layer of information systems. It interprets knowledge queries from its user and presents knowledge in a context-adequate accessible form. *Information agents* retrieve, filter, and accumulate knowledge either from large unordered stocks of data (the world wide web, mail boxes, data bases, etc.) or from organisational memories. *OntoBroker agents* exploit the ontology-based structure of a non-agent-based organisational memory for information retrieval (Staab & Schnurr 2000). A

reactive version of the agent requires a comprehensive specification of the user's business context while the proactive version employs heuristics to infer missing context pieces and is equipped with mechanisms for handling potentially huge numbers of relevant queries. Rhodes and Maes have presented three just-in-time information retrieval agents (JITIR) (Rhodes & Maes 2000). The *Remembrance Agent* continuously presents a list of documents related to that currently written or edited in the emacs editor. *Margin Notes* uses web documents as context to draw the user attention to related personal information items. *Jimminy* uses clues from the physical environment of the user such as her location or proximity to further persons to determine what information may be relevant at present. At MIT, the *Letizia* interface agent has been developed which assists a user browsing the web so as to further knowledge capitalisation and reuse (Rhodes & Maes 2000). To do so, Letizia passively observes the user and learns a preference profile. This profile can then subsequently be employed to anticipate and tend to future information needs of the user.

With exceptions such as Letizia, agents for personal knowledge management adopt the agent metaphor, yet are typically implemented with conventional technologies. While heavily drawing on various techniques for information retrieval such as ontology-based access to formalised knowledge items, van Elst et al. remark that research has focussed primarily on deeply embedding the agents in their virtual/physical environments, i.e., sensors for context awareness and effectors for information presentation.

Recently, with the fast-paced advancement of mobile computing in the smartphone and tablet space, specifically the growing availability of hardware sensors, and access to powerful cloud-based services, mobile assistants have entered the mainstream with popular services such as Apple's Siri[1] or Google Now[2].

**Leveraging Knowledge Distribution** Advancing on the continuum of sociability (see Figure 3.2), predominantly homogeneous multiagent systems have been considered. For instance, MARS has been proposed as an adaptive social network for information access (Yu et al. 2003) which leverages the idea that any user in the network can essentially both assume the role of a knowledge consumer and knowledge provider. Hence, it is acknowledged as previously worked out in Section 3.1.2 that knowledge is inherently distributed in organisations and societies and that knowledge transfer processes (Section 3.1.3) can serve to establish channels to tap such distributed expertise directly rather than through the indirection of corporate memories. Each MARS agent is basically endowed with two competencies:

a) to deliver some domain information with respect to a client query, and
b) to refer to fellow agents that are deemed capable to accommodate specific information needs.

During operation, the agents also engage in learning peer profiles. Once constructed, these provide means to assess the expected quality of service with regard to answer generation and peer referral.

1 See `http://www.apple.com/ios/siri/` (visited: 2012/11/20)
2 See `http://www.google.com/landing/now/` (visited 2012/11/20)

In Chen et al. (2000), DIAMS has been proposed as a system of collaborative information agents designed to facilitate access, collection, organisation, and transfer of information on the world wide web. Personal information agents provide their respective user with dynamic views on well-organised local information repositories. A DIAMS user can visit the repositories of other users through his/her personal agent and it is possible to include structured information objects from external repositories in one's own collection. Access to other users' repositories is done through "behind the scene" categories search and translation between agents. When a personal agent refers a user query to a peer, additional meta-information with regard to the local query accommodation is included. The queried agent is endowed with user-configurable policies which allow, for instance, for an update to its own local repository. DIAMS is 'weakly homogeneous' in structure as it employs additional facilitator agents. Most notably amongst them are matchmakers which provide yellow page services and can thus serve as point of reference for the discovery of new personal agents in the system.

**Comprehensive Agent-based Knowledge Environments** On the high end of the sociability spectrum proposed in (van Elst et al. 2004), systems like DIAMS materialise the trend towards large-scale heterogeneous multiagent knowledge management environments. Not only do such systems comprise a potentially high number of agents, but the division of labor in the system increases such that many different agent types with specific competencies and goals contribute to system operation. Heterogeneity can either be due to large numbers of real-world entities which are reflected in the system, functional decomposition in the system design, or it can be a consequence of the support of a wide spectrum of knowledge management processes compared to the 'weakly homogeneous' systems discussed thus far.

For a knowledge reuse-oriented view, the integration of information from various sources is a central challenge, addressed as semantic mediation (Wiederhold & Genesereth 1997) or semantic interoperability. A first project which focussed specifically on the fusion of knowledge from multiple, distributed and heterogeneous sources is expressed in the Knowledge Reuse and Fusion/Transformation (KRAFT) architecture (Preece et al. 2000, 2001). Within this architecture, constraints are used as a common knowledge interchange format whose concepts are specified in terms of a shared ontology. KRAFT uses an open and extensible agent architecture in which various knowledge sources, entities that perform knowledge fusion, and clients that draw on the services of the system are represented as software agents. Additional facilitator agents perform semantic matchmaking and brokerage services. Two basic operations are supported by KRAFT. For *knowledge retrieval*, a knowledge fusion is used to aggregate all relevant information, specifically associated knowledge items and constraints as meta-information, across all knowledge sources in the environment. These are processed into a consolidated representation format. *Problem solving* builds on these aforementioned capabilities and uses the aggregated information as basis for a dynamically constructed constraint satisfaction problem. Hence, KRAFT lends itself, amongst others, for solving configuration problems.

Preece et al. (2000, p. 114) pose that the essential services to be procured by a knowledge fusion system comprise

**Figure 3.3:** Case diagram of user request processing in KSNet (Smirnov et al. 2002, p. 300)

a) *knowledge location services*,
b) *knowledge transformation services* to homogenise representation language and ontology for handled knowledge items, and
c) *knowledge fusion services* which essentially combine and process knowledge.

Hence, with respect to the classification of knowledge management processes by Alavi and Leidner (Alavi & Leidner 2001) (Section 3.1.3), KRAFT integrates knowledge retrieval with knowledge creation processes, for the latter focussing on the combination of various sources of explicit knowledge. KRAFT has been implemented as a FIPA-compliant multiagent system and has been applied in network data services design and for advertisement of students for university transfers.

Beyond KRAFT, Smirnov and colleagues have presented a further multiagent architecture for knowledge fusion under the name of KSNet, an acronym for Knowledge Source Network (Smirnov et al. 2002). The authors emphasise the importance of engineering a shared ontology at system design time. The division of labour among agents is even more pronounced than in the KRAFT project (see Figure 3.3) such that queries by a user of the system lead to a long process chain involving very heterogeneous agent stakeholders including user agents, mediator agents, translation agents, ontology management agents, knowledge fusion agents, just to name a few. Only their diligent orchestration enables the distributed system to procure its knowledge management services.

Somewhat related to knowledge fusion, Ontañón and Plaza have recently proposed an argumentation-based approach to multiagent inductive learning which can also be viewed as a contribution to agent-mediated knowledge management (Ontañón & Plaza 2010*d*,*b*). Within the framework of concept learning, agents endowed with learning capabilities first individually induce classification models for a concept learning task based only on a local data source. However, the objective of the considered agent society is the fusion of these local models into a single global

version. Before the contributing models can be integrated (this again, refers to the combination mode of knowledge creation), conflicts in the source models need to be addressed. This is achieved by means of formal argumentation (Bench-Capon & Dunne 2007) in that the agents present their respective models to each other and have disputable rules challenged by peers. By argumentation (challenge) and counter-argumentation (defence), it is determined whether models need to be adapted in order to achieve mutual agreement.

Changing focus from knowledge fusion systems, it can be stated that in an organisational environment, an important context facet of organisation members engaged in knowledge intensive tasks is the embedding in a specific business process which shapes knowledge needs. *Business process-oriented knowledge management* (Abecker et al. 2002) hence accommodates these processes

a) as items which are themselves represented in a knowledge management system,
b) as context for knowledge processes such as creation,
c) as trigger *when* specific knowledge becomes relevant, and
d) *which* knowledge is then relevant (van Elst et al. 2004, p. 18).

In the FRODO framework which implements a distributed organisational memory, the workflows themselves are first-order citizens in an agent society for knowledge management in distributed environments (Abecker et al. 2003). Abecker et al. content that an explicit modelling of business processes as a means for context representation facilitates context-aware information retrieval and the proactive procurement of relevant information. The authors propose to build their distributed organisational memory as a set of collaborating societies of socially enabled agents. An *agent society* is thereby defined as a collection of agents including at least a single management agent (administering society membership, role assignments and the like) which enact for a limited time one or more *agent roles* with respect to this society. A role within a society is thereby constituted by its associated rights and obligations. Both concepts describe a subset of an agent's competencies. The rights define under which circumstances an agent is allowed to take action, while obligations define circumstances where taking action is specifically asked for. Agents that enact roles within a task-focussed society are referred to as socially-enabled agents. These agents have been realised in FRODO on top of the JADE platform (Bellifemine et al. 2007).

FRODO comprises several interacting societies which jointly provide knowledge management services. A first society is dedicated to ontology management (Abecker et al. 2003, p. 6). Rather than trying to engineer and maintain a single shared ontology as in the KRAFT project (Preece et al. 2000), FRODO acknowledges that different partially autonomous departments or user groups in an organization will realistically maintain their own ontologies. Hence the role of a distributed domain ontology agent is introduced which acts as mediator between agent societies represented by respective domain ontology agents. A second agent society handles the representation and enactment of business processes. Dedicated agents have been implemented to represent business processes, tasks therein, and process-critical resources. FRODO also implements personal user agents. Finally, at the core of the FRODO system is the information processing agent society. Herein, so-called info agents care for information retrieval from multiple data

sources which may be plugged into the system and context providers which can advise as to which context facets may be helpful in improving specific information processing tasks.

Another approach to distributed knowledge management has been investigated in the Edamok project (Bonifacio, Bouquet, Mameli & Nori 2002) with the peer-to-peer based architecture KEx. Each peer in KEx has the competence to create and organise knowledge which belongs to an individual or a distinct group within an organisation. Social structures between peers are constructed which then enable knowledge exchange. As with the previous approaches, semantic coordination techniques and contextual reasoning are investigated in Edamok. An approach which is closely related to both FRODO and Edamok has been developed in the CoMMA project (Bergenti et al. 2000). The authors act on the suggestions from Abecker et al. with respect to the modelling of agent societies within their architecture for personal information delivery.

With OperA, Dignum has proposed a comprehensive design methodology which is specifically tailored to agent societies (Dignum 2004). The methodology is based on a layered three-part framework for agent societies. It distinguishes between the specification of a desired organisational structure (macro-model) and individual agents with their respective goals and behaviours. On the top-level, the organisational model reflects the intended organisational structure of the agent society, as intended by organisational stakeholders. One level below, the concrete agent population of an organisational model is specified in the social model. This is done in the form of contracts as explicit commitments concerning the enactment of roles by individual agents. Finally, given an agent population which fills the organisational model with life, the interaction model constrains possible interactions among the agents. The OperA approach stresses the importance of a thorough analysis of the processes, stakeholders, their knowledge needs and interrelationships for a particular application domain. On this basis, a suitable projection onto an adequate agent-based knowledge management environment can then be carefully devised. While the agents in Dignum's approach are granted with autonomy with respect to the conduct of apportioned roles, the approach nevertheless stresses careful top-down engineering.

To conclude the exemplary description of comprehensive agent-based knowledge management environments, it has been shown that the breath and methodical diversity required to construct large-scale multiagent systems for knowledge management is considerable. This is especially true when new challenges for knowledge management systems are addressed explicitly, such as the deep integration with business process workflows or the enabling of semantic interoperability in distributed heterogeneous environments. As a consequence, the number of complementary competencies which must be exhibited by considered agents increases and a trend can be observed towards specialisation and distribution of work. Modelling of multiagent systems in terms of agent societies and the role-paradigm have been proposed to cope with the new design challenges.

## 3.4 Agent-oriented Knowledge Management

In reviewing the objectives of knowledge management (Section 3.1.1) and its constituent knowledge processes (Section 3.1.3), it is rendered evident that from its beginning, the workforce of an organisation, its human decision makers concerned with knowledge-intensive tasks within organisational business process, were at the core of knowledge management initiatives. Reenacting the evolution of knowledge management systems from early centralised approaches towards comprehensive frameworks for agent-mediated knowledge management (Sections 3.2 and 3.3), the role of more and more intelligent information technology as critical *enabler* and *facilitator* for organisational and individual information and learning has been illustrated. This observation is for instance supported by Dignum who states with regard to agent-mediated knowledge management that the use of agent technology in knowledge management can be seen from two perspectives. First, agents can be used to model the organisational environment where a knowledge management system is to be deployed. Second, and this is the predominant perspective, software agents can be seen as implementation tool. (Dignum 2006, p. 179).

However, in order to cope with the growing complexity and dynamics of modern business processes, illustrated by the rise of virtual organisations or in the logistic area by large-scale supply networks, it has been recognised that knowledge management systems themselves need to exhibit core qualities of intelligent agents, namely autonomy, reactive and proactive behaviour and especially in distributed and heterogeneous settings social ability.

Yet, even when respective systems act towards their users as holonic entities with agent characteristics, several decisive elements that drive the effective implementation of knowledge management reside with the human decision maker as client of knowledge services. Paradigmatic examples include situation assessment as to which missing pieces of information are truly relevant and beneficial in a current business context. The most critical example, however, are the actual learning processes by which the knowledge worker expands her individual knowledge in the context of knowledge creation processes (Section 3.1.3), including socialisation and internalisation.

**Implications of Autonomous Control for Knowledge Management** The complementary task sharing among knowledge workers as handlers of business processes and, in this capacity clients of agent-mediated knowledge management environments on the one hand, and agents as handlers of interconnected knowledge processes on the other is challenged by the paradigm shift towards autonomous control. Subject to numerous research activities first in autonomous control of logistics processes (Freitag et al. 2004) and more recently cyberphysical systems, the paradigm promotes the renunciation, or reduction of dependency on centralised planning and control in favour of heterarchical networks of interacting nodes with local decision making competencies, each better attuned to subtasks in superordinate business processes. An essential characteristic of this transformation is its emphasis on the inclusion of technical systems as first-class

citizens in the group of decision makers among whom the responsibility for the handling of business processes is divided. Hence, the proposed transition to autonomous control is to disburden highly-trained human decision makers as a growing number of business processes can be handled autonomously by digital representatives which can be adequately realised in terms of intelligent agents (Gehrke et al. 2010).

While the development of novel planning and control methods for autonomous control poses an active area of research (Schuldt 2011), challenges also arise for knowledge management. Since agents as digital representatives are endowed with knowledge-intensive tasks that previously resided within the responsibility of human decision makers, and need to act in a highly-distributed environment subject to frequent reconfiguration, knowledge management *on behalf* of the agents themselves becomes an essential factor for effective conduct of the agents' business-related objectives. The accommodation of agents as stakeholders in knowledge management processes against the background of their own process-driven knowledge needs in the author's view goes beyond the scope of agent-mediated knowledge management presented in the preceding section. For the scope of this thesis, we hence use the term agent-oriented knowledge management[3]. This emphasises the agents as focal point of knowledge management which is motivated and autonomously conducted by software agents engaged in autonomous control.

**Towards Agent-oriented Knowledge Management** In several publications, Langer and colleagues have proposed a generic framework for distributed knowledge management in autonomous logistics processes (Langer et al. 2005, 2006, 2007). The main contribution is the classification of knowledge management functions for agent-oriented knowledge management and a role-concept for the flexible, time-variant mapping of these functions to agents. In contrast to the agent-based approaches presented in Section 3.3, the authors consider knowledge management as auxiliary processes in service of the primary agent tasks to handle business processes which arise in supply chain management, transport logistics, or production. While for instance FRODO does accommodate business processes as first class-citizens in their implementation, the rationale for such business process-oriented knowledge management is the opportunity for a rich context representation facilitating focused information retrieval and pro-active information procurement. Using their proposed framework, Langer and colleagues rather seek a tight integration of planning and control on the operational side and knowledge management on the other side. It is stated that "*knowledge management as it is proposed in this framework is one key enabling factor to the envisioned autonomy of logistic processes. Autonomous entities need to make decisions based on technically implemented decision-theoretic processes. In order to achieve this, they not only need knowledge about their environment but also have to assess future states of the environment and judge alternative options*"(Langer et al. 2006, p. 282).

Also, despite potentially existing acquaintance of different agents due to their respective organisational affiliation, the authors refrain from requiring initial

3 The presented notion of agent-oriented knowledge management is not to be confused with agent-oriented knowledge management presented by Guizzardi (Guizzardi 2006) which focusses clearly on support of human knowledge workers.

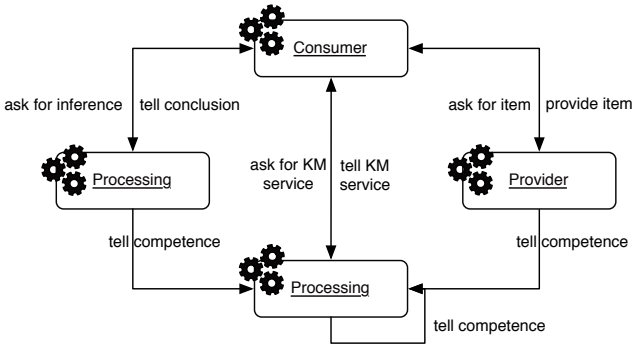



**Figure 3.4:** Inter-agent role-oriented communication acts (adapted from Langer et al. 2006, p. 283)

infrastructure in the knowledge management network such as a centralised organisational memory. This is consistent both with the inherently distributed nature of knowledge in organisations (Section 3.1.2) that is reflected in distributed and and agent-mediated knowledge management research (Sections 3.2.2-3.3). Langer et al. stress the necessity for a dynamic, situation-dependent adoption of different knowledge management roles such that the structure of the knowledge management environment emerges from the interplay of complementary roles following the principle of self-organisation. This bottom-up approach differs from that represented, for instance, by the OperA model (Dignum 2004). However, it is a consequence of autonomous control.

The system of generic knowledge management roles which has been proposed is shown in the Figures 3.4 and 3.5. The authors stress as unique feature of their approach that no fixed one-to-one correspondence between specific agents and those knowledge management functions encapsulated in roles is predetermined. While throughout Section 3.3 the general trend pointed to a diligent orchestration of various task specialists, amongst others for meditation services, ontology management, translation (see Figure 3.3), this new approach favours flexible agents with a rich repertoire of knowledge management capabilities to be adopted based on their respective needs and perceived ability to provide value-added services to peers. The role metaphor adopted by Langer et al. is derived from a sociological role concept (Herrmann et al. 2004) rather than an organisational modelling primitive as in (Dignum 2004). In a case study on a cooperative learning environment Herrmann et al. have investigated the dynamic adoption of roles by human learners. These roles then explained how actors interact, collaborate, and work together to cultivate knowledge exchange.

The notion of roles proposed by Langer et al. includes

a) certain embedded reasoning capabilities,
b) a visibility function on an agent belief base,
c) a deliberation pattern informally described as a plan on how to accomplish the modelled knowledge management function, and finally
d) a communication behaviour with interacting roles.

Parallels exist to the role concept for socially enabled agents in the FRODO project (Abecker et al. 2003). For instance, the visibility function could be expressed as a *right* to access and certain information while the role *obligations* to act must

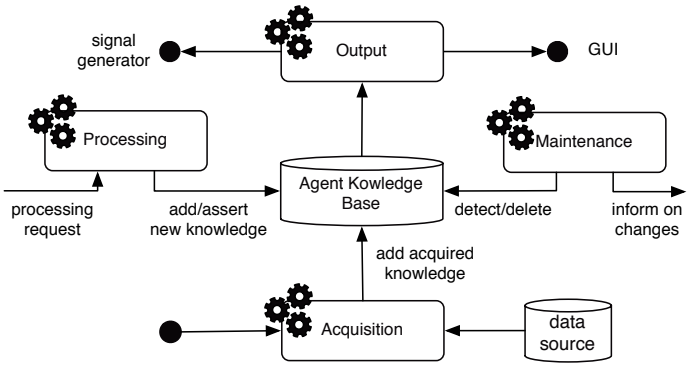



**Figure 3.5:** Functions of internal knowledge management roles with respect to the agent knowledge base (adapted from Langer et al. 2006, p. 283)

be modelled in the deliberation pattern. The communication behaviour with interacting roles finds its correspondence in the interaction model which is part of the OperA modelling framework (Dignum 2004).

Langer et al. distinguish internal roles (Figure 3.5) and external roles (Figure 3.4). The latter roles have an interactive character in that they have to be specified in initiator/responder pairs within an instantiated role system. Respective role interactions have been highlighted in Figure 3.4. The proposed external role templates comprise those of immediate stakeholders in knowledge transfers; the *processing* and *provider* role on the offering side, and the *consumer* on the demand side. With the *broker* role, the requirement for additional facilitator roles is accommodated. With specialisations of these templates (see, e.g., Section 5.2), knowledge processes for creation, retrieval/storage, transfer, and application of knowledge (Section 3.1.3) can be modelled. The internal roles that have been defined with respect to the knowledge base of an agent model local knowledge handling capabilities. On the one hand, they comprise the direct acquisition of information from the environment and the presentation of views on the agent knowledge base for a user. On the other hand background processes classified as *processing* and *maintenance* handle the agent-internal management of its knowledge base.

Specific functions within the proposed knowledge management framework have already been investigated in previous research conducted primarily in the context of the Collaborative Research Centre 637 on Autonomous Cooperating Logistic Processes (Freitag et al. 2004). Hribernik, Kramer, Hans & Thoben (2010) have developed a semantic mediator for application in autonomous logistics which can be seen as an instantiation of the knowledge processing role due to its ability to translate between different information schemes. The mediator internally builds upon formal ontologies which have been engineered specifically for the modelling of logistic processes (Porzel & Warden 2010, Warden et al. 2010). In several publications, the internal knowledge processing role has been instantiated allowing agents to learn models for situation assessment from previously gathered experience (Gehrke & Wojtusiak 2008*b*), or perform knowledge-intensive planning of container shipping (Wojtusiak et al. 2012*b*, Warden et al. 2012). Gehrke (2011) has investigated the assessment and optimal acquisition of thus far unknown pieces of information based on the information value theory.

To conclude, important steps towards agent-oriented knowledge management have been taken. Rather than replicating the scope of comprehensive agent-mediated knowledge management approaches (Section 3.3), researchers have thereby concentrated on complementary knowledge processes aligned to the domain of autonomous logistic processes. However, specifically Gehrke's work on information acquisition for the decision making of utility-based agents points to a further research requirement partly addressed in this thesis: The mapping of cognitive processes normally associated with human clients of a knowledge management system onto software agents. While Gehrke endowed the agent with the ability for informed acquisition of information in the context of a concrete decision situation, the next step is the support of more strategic knowledge processes. Specifically, agents need to be enabled to participate in processes of knowledge creation, thereby deriving surplus value from already existing local knowledge.

## 3.5 Chapter Summary

The chapter has described the role of agent-based modelling and multiagent technology for the multidisciplinary field of knowledge management. In order to provide a solid foundation, Section 3.1 has outlined both scope and objective of knowledge management from an organisational point of view, thereby acknowledging the roots of the field in management theory. Due to its centrality for the topic, important perspective views on knowledge and related concepts such as information and data have been introduced. Thereby, it became clear that the respective conceptualisation of knowledge, for instance knowledge as an object that can be codified, stored and accumulated in a repository versus knowledge as tacit concept which necessarily resides within an agent, significantly shapes the perception of the mission of knowledge management. Section 3.1.3 rounded up the constituent processes which constitute building blocks for knowledge management in organisations and beyond. With respect to the objective of this thesis, processes contributing to knowledge creation with individual members of an organisation or society, based on relevant knowledge that is hold available in the collective, are of particular interest.

Section 3.2 acknowledged the important role of information technology as enabler for knowledge management activities. Specifically the evolution of knowledge management systems from centralised systems towards distributed systems has been outlined, a paradigm shift which acknowledged the inherently distributed nature of knowledge in organisations. Section 3.3 outlined the growing significance of multiagent technology for the realisation of agent-mediated knowledge management systems. These have been shown to range from single-agent approaches, i.e., personal information agents, up to comprehensive frameworks such as FRODO which cover a wide spectrum of knowledge processes and build upon large ensembles of specialised agents to jointly procure the desired functions to human clients.

Finally, Section 3.4 takes on the paradigm shift envisioned in the principle of autonomous control and carves out its implications for knowledge management. The

principal finding here is that the transformation from agents as (business) process companions and digital assistants of knowledge workers towards process owners broadens the scope of knowledge processes handled by technically implemented knowledge management. Where in agent-mediated knowledge management it sufficed to create the conditions for individual learning of knowledge workers, autonomous control necessitates the conferment of such interactive learning capacities to software agents. This requirement is the incentive for the subsequent chapter to investigate machine learning methodologies with a specific focus on active learning with and from peers (Section 4.2). Drawing on such contributions, the scope of agent-oriented knowledge management is to be extended with respect to knowledge creation in the context of individually learned classification models for situation assessment and prediction.

Chapter 4

# Machine Learning in Multiagent Systems

In complex task environments, actionable knowledge of characteristics of this environment and other stakeholders therein has been identified as a critical factor for the competitiveness of intelligent agents in multiagent systems. The review of common agent architectures in Section 2.2 highlighted methods to found agent behaviour on codified knowledge. Also, the potential to improve upon an initial state of knowledge through integration of performance and learning element has been established (cf. Section 2.3). It has been argued in Section 2.4 that the individual agent is often part of a community of practice, a network of other agents either from a common organisational context or a non-adversarial third party with similar tasks, incentives, and again knowledge to support their actions.

Knowledge management acknowledges that actionable knowledge in the individual constitutes a valuable asset and that its distribution in various qualities and codifications presents both an enormous challenge and potential to create added value in integrating knowledge sources and draw new deductions (cf. Section 3.1). Research in knowledge management has characterised the structure of knowledge networks, and qualified stakeholder roles and knowledge management functions whose interaction is essential for information exchange aiming at knowledge refinement. Agent-oriented knowledge management investigates the formation of dynamic knowledge networks in multiagent systems where knowledge management is performed by agents for agents (cf. Section 3.4).

Hence, multiagent system applications yield the incentive for agent-oriented knowledge management. Multi-agent research procures the enabling technologies to emerge knowledge networks among intelligent agents.

This chapter consequently investigates machine learning techniques to fill the *knowledge processing* role that is integral to individual *knowledge creation*. Specifically, the review focusses on techniques in which the learning systems already exhibit agent-like characteristics in that they assume an active role not only in information processing but also goal-oriented knowledge acquisition from third parties to shape the learning process.

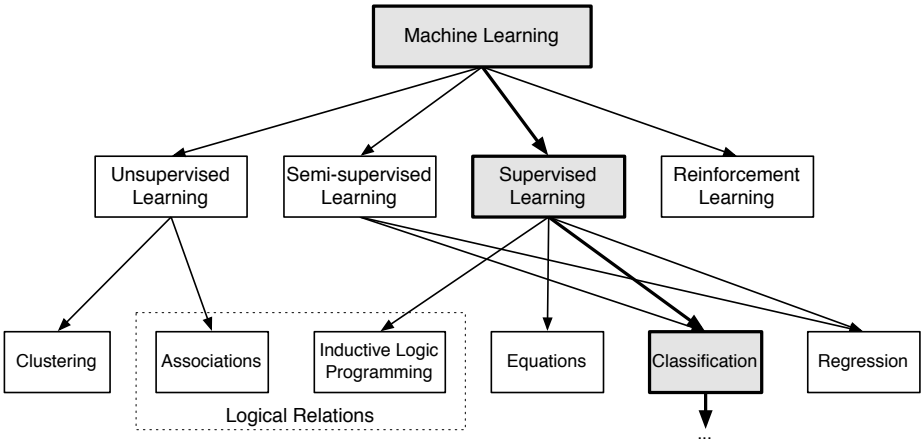



**Figure 4.1:** A taxonomy of machine learning tasks, adapted from Kononenko & Kukar (2007, p. 6). Tasks to be discussed in more detail in the following sections are greyed.

The following section sketches a landscape of machine learning techniques in traditional non-activated scenarios.

Section 4.2 is then concerned with variations of this fundamental learning setup. Specifically, *active learning* retains the focus on a single-learner but deviates from the hitherto rendered assumption that whatever input data is available for the momentary learning task is invariant. The basic theme is that the learner is endowed with an active role within data acquisition. To enable such proactive behaviour reminiscent of the goal-directed behaviour of a software agent, the actual learning process is broken into episodes, interleaved with acquisition of additional data. The data acquisition itself can thereby take various forms ranging from the autonomous conduct of (physical) experiments to interaction with human domain experts.

Section 4.2.2 introduces *argument-based learning* which differs from active learning as the learner seeks clarifying advice on the classification of known instances from advisors. Section 4.2.3 continues with *semi-supervised learning*.

In its second part, the focus of the survey is shifted to learning in multi-agent environments. First, the case of single-agent learning considered thus far is contrasted with its multi-agent analog. A categorisation of multi-agent learning variants is introduced, ranging from multiplied to distributed and finally interacting learning. The text will discuss the predominant motivations for agent learning. Finally, the text considers learning setups in which, analogous to active learning, the learning agents proactively seek assistance from other knowledgeable peers in order to enhance their own learning processes.

## 4.1 A Landscape of Machine Learning Methods

Acknowledging the breadth of the machine learning field with respect to areas of application and methodologies, the taxonomy of machine learning branches shown in Figure 4.1 and adapted from Kononenko & Kukar (2007, p. 6) serves as

a point of reference.

Probably the most widespread field in the partition of the machine learning landscape is that of *supervised learning*. The common denominator for supervised learning is found in the composition of source material for learning. Specifically, the learning input is composed of a collection of *cases* $s_i = \langle \mathbf{x}, y \rangle \in I^{dom}$ from a particular task domain. In a job application scenario, a case $s_i \in I^{job}$ might be the profile of an applicant, characteristics of the employer, together with the decision about the employment decision. In an accident injury scenario, a case $s_j \in I^{acc}$ might be the detailed incident report from the authorities and applied medical treatment such as first aid measures or hospitalisation, together with the actual injury severity. The supervision aspect in supervised learning derives from the fact that each case description is composed of a *feature vector* $\mathbf{x}$ where each dimension $x_i \in \mathcal{F}_{num} \cup \mathcal{F}_{nom}$ comprises a specific case characteristic, and a separate scalar *target feature* $y \in \mathcal{F}_{num} \cup \mathcal{F}_{nom}$. $\mathcal{F}_{nom}$ thereby denotes the set of qualitative, i.e., discrete nominal features, while $\mathcal{F}_{num}$ denotes features whose values lie on a continuous scale. In supervised learning, the understanding is that the feature vector for an instance represents a set of observable independent variables that effectuate the value of the preset dependent target variable.

The goal of supervised learning is the deduction of predictors that implement a dependable function $\texttt{pred} : \mathcal{F}^i \ldots \mathcal{F}^n \mapsto \mathcal{F}^{target}$ to find values for the dependent variable on previously unseen, unlabeled cases $s_i = \langle \mathbf{x}, ? \rangle$, given a collection of labeled cases $I_{train}$ as training data. The model that is induced should be

a) concise,
b) reveal structural patterns in the training data, and
c) should translate well to novel cases rather than only summarise seen data.

As seen in Figure 4.1, supervised learning can be further divided along dimensions such as the nature of the target feature. Those supervised learning tasks for which the target feature is discrete, i.e., $y \in \mathcal{F}_{nom}$, are denoted as *classification* tasks and the respective predictors are called *classifiers*. If, by contrast, the target feature is continuous-valued, i.e., $y \in \mathcal{F} \in \mathcal{F}_{num}$, one speaks of *regression* tasks. A special sub-field of supervised learning is learning systems of *equations*.

Before focussing onto classification in more detail (cf. Figure 4.2), as the thesis at hand will employ classifiers as decision support models for learning agents, the delineation of the other machine learning subfields in Figure 4.1, most notably *unsupervised learning* and *reinforcement learning*, is established.

Unsupervised learning is different from the supervised scenario given in classification and regression since the individual cases that still form the source material for learning only comprise a feature vector without a dedicated target feature such that $s_i = \langle x \rangle : x_i \in F^i \in \mathcal{F}_{num} \cup \mathcal{F}_{nom}$. *Association learning* constitutes a branch of unsupervised learning which defines the learning goal to find interesting and relevant structural patterns among the constituent elements of the feature vector. Conceptually, association learning, missing guidance through supervision, corresponds to $n$ classification/regression tasks where $|\mathbf{x}| = n$.

The goal of *clustering* algorithms, on the other hand, is a partition of the input data into meaningful sub-sets based on a distance metric that is applicable to the cases $s_i = \langle \mathbf{x} \rangle \in I_{train}$. Depending on the respective algorithm, the concrete

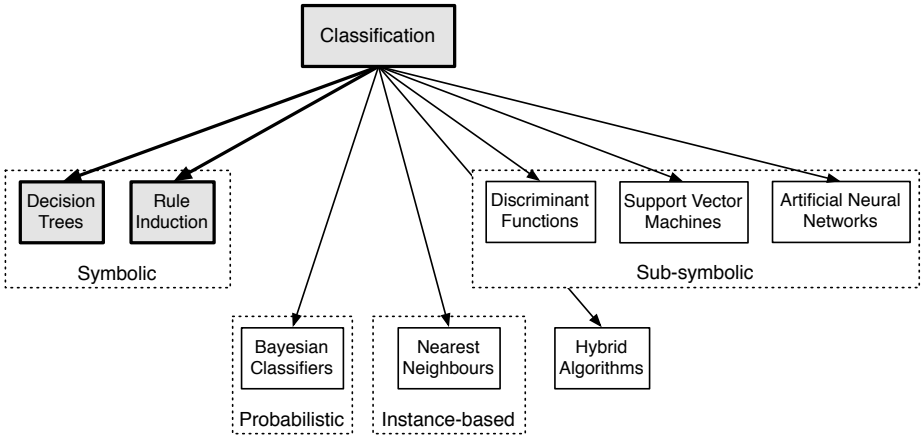



**Figure 4.2:** A taxonomy of classification methods, adapted from Kononenko and Kukar (Kononenko & Kukar 2007, p. 6). Methods that are utilised in this thesis and are discussed in subsequent sections are greyed.

number of clusters is either determined by the algorithm itself or preset by the user.

*Semi-supervised learning* constitutes an extension of supervised learning. Starting from the same initial situation as supervised learning, semi-supervised approaches differ in that the learners are enabled to take an active role in their data acquisition and hence can shape their own learning process. Section 4.2 provides details on this machine learning branch.

Finally, reinforcement learning deals with the problem of teaching an autonomous agent that senses its surrounding environment with sensors and acts upon entities in the environment with mostly physical actuators an action selection policy that maximises goal achievement or utility. The agent receives information about the current state of the environment and performs actions to change it. Actions effectuate the reception of a positive (encouragement) or negative (punishment) reward. However, the rewards can be the product of a possibly very long series of actions. Due to such reward delays learning is extensive. The learner needs to strike a balance between exploitation of existing knowledge on successful actions with exploration of such actions and action sequences whose result is still unknown. The task of the learner is to maximise the cumulative reward over time. When working toward discrete goals, the fastest approach is thereby often deemed best, which is accounted for in the reward calculation.

Having discussed the top-level partitioning of the vast machine learning field, the focus is narrowed down to supervised learning and more specifically to classification.

Figure 4.2 constitutes a cutout refinement of the taxonomy in Figure 4.1 on page 62 as it clusters well-known classification approaches based on the representation language that is used for trained classification models.

A first distinction involves whether or not the respective approach distinguishes the compilation of a classifier in a dedicated training phase within the learning life cycle. Here, *instance-based learning* is idiosyncratic in leaving out a dedicated training phase aimed at model creation that precedes an eventual operative

exploitation of the model for the classification of novel unlabeled cases. The set of training data itself is retained as a basis for concrete classification queries. It is thus not substituted by a model abstraction in any representational language. As a consequence, this model-free approach entails complexity in the computation of answers to concrete classification queries. Specifically, such a lazy classification typically entails the heuristic evaluation of an instance neighbourhood based on a domain-specific distance metric among instances.

The dichotomous class of classification approaches is distinguished by the employment of a dedicated classification model that codifies domain knowledge deduced in a training phase and its exploitation in servicing classification queries. Model induction is thereby typically a computationally intensive task while model exploitation is cheap and fast. The respective model-based approaches have as common denominator that they allow their owners to forget concrete past cases while still memorising the gist, i.e., significant structural patterns in these cases in a compact representation.

The representational language to express the respective classification model serves as a criterion to partition the model-based classifiers into three groups: 1) symbolic representations, such as decisions trees and propositional rule bases, 2) subsymbolic representation, most prominently among them discriminant functions, support vector machines and artificial neural networks, and finally 3) probabilistic representations which include Bayesian networks.

The adoption of one of the aforementioned families of model representations can be a conscious decision in a specific application scenario, or it can be a consequence of a design choice that is rendered based on the expected performance of a trained classifier measured in terms of quality-of-prediction key performance indicators.

### 4.1.1 Symbolic Classification Techniques

Symbolic and, at least for the experienced knowledge engineer, probabilistic techniques yield what might be referred to as white-box models. That is to say, that such models not only act as means to the end of answering classification queries. They also afford the direct inspection of a trained model as the structural patterns captured in the model are intelligible to human stakeholders. Besides comprehensible advantages for application development and traceability, white-box models render accessible interesting properties of the task environment. Propositional rule bases compiled by rule induction algorithms consist of discrete rules that each exhibit a semantical structure that is very close to sentences in natural language. Single chunks of knowledge are hence immediately comprehensible, although the overall picture may be hard to comprehend in case of a large rule base. Decision trees, as a related propositional representation, by contrast feature an easily traversable tree structure that encodes an ordering of the features in a feature vector for a task domain in addition to encode rule-like expressions as paths from the root of the tree to its leaves.

Trained Bayesian Networks yield information about causal relations and their strengths between features in the task domain and the dependent variable. Bayesian networks have also been considered as points of origin for the extraction

of symbolic models, as in (Hruschka et al. 2007, Nielsen et al. 2008).

To conclude, symbolic and to some degree probabilistic representations induce comprehensible classification models that render explicit interesting pieces of domain knowledge. Section 4.2.2 even argues that the respective representations, specifically propositional rules, afford sophisticated discourses on selected, often not universally accepted, fragments of a classification model.

### 4.1.2 Sub-symbolic Classification Techniques

In contrast to the symbolic family of classification techniques, their so-called sub-symbolic complements feature as common denominator an often opaque, black-box knowledge representation. Prominent types of sub-symbolic classification techniques, as shown in Figure 4.2 include1) discriminant functions, 2) Support Vector Machines, and 3) Artificial Neural Networks, including, in particular, the deep neural network approaches investigated in the Deep Learning field.

With discriminant functions, the task of a respective learning algorithm is the optimal assignment for the coefficients of the eponymous function whose structure is typically preset as an additional input to the learner besides a training data set. A discriminant function is actually a hyper-surface whose margin is to separate cases of one particular target class from those of all other classes with minimum error. A classification target with a multi-valued, i.e., non-binary domain is supported with one hyper-surface per class and hence dichotomy. The concrete discriminant functions can be amongst others linear, quadratic, polynomial, with Fisher's linear discriminant function (Fisher 1936) being a popular choice in practice.

Support Vector Machines (SVMs) constitute a further development on discriminant functions introduced in the 1990s (Cristianini et al. 2000). They are a multi-criterion optimisation method. Summarised briefly, SVMs maximise the distance between support vectors and a dichotomising hyper-plane where support vectors are those examples closest to the hyper-plane thus defining a margin. Hence SVMs maximise said margin as part of the first optimisation criterion. Due to missing linear separability of many classification problems, SVMs also implicitly transform the original feature space into an equivalent space of much higher dimension using an appropriate Kernel function. Its choice is critical. Compactness, in terms of necessary coefficients is also a criterion. The number of coefficients is subject to minimisation, as well as the residual classification error due to nonlinearities. While the success of SVMs for classification is well-documented, they constitute also a paradigmatic example of a black-box knowledge representation that is well-suited to answer new classification queries. However, it is hard to draw additional inferences about the domain itself or explanations on classification decisions.

Artificial Neural Network (ANNs) (Rojas 2013) comprise another sub-symbolic family of classification techniques. Although the field originated in the cybernetics research in the 1940s–1960s, and has seen renewed interest in the connectionism of the 1980s (cf. (Goodfellow et al. 2016, pp. 13)), it has recently seen a surge of interest with considerable breakthroughs in the Deep Learning field (Goodfellow

et al. 2016). Initially inspired by the powerful natural signal processing in human and animal brains, ANNs emulate biological neural networks by abstracting complex neural functions to networks of individually simple, yet highly interconnected functional elements (neurons) that summarise stimuli on their weighted inputs and normalise their outputs.

For classification purposes, multi-layered feedforward neural networks are employed, whose input neurons correspond to the elements of the input vector and whose output layer corresponds to the distinct possible values in the domain of the target feature. A feedforward neural network hence defines a mapping $y = f(x, \theta)$. Similar to SVMs, the learning task with for such a network comprises the determination of suitable connection strengths $\theta_i$ among the individual neurons. Convolutional Neural Networks (CNNs) (LeCun et al. 2010) are specialised feedforward networks for object recognition tasks. Recurrent Neural Networks (RNNs) (Graves 2012) extend feedforward networks with feedback connections in order to process time-series data, for instance in speech recognition.

According to Goodfellow et al. (2016, p. 4) Deep Learning constitutes a form of representation learning where the machine learning task comprises not only the learning of a feature mapping but also the determination of an appropriate feature vector. Specifically, the potentially large number of hidden layers in deep neural networks enable the representation of the world "*as a nested hierarchy of concepts, with each concept defined in relation to simpler concepts, and more abstract representations computed in terms of less abstract ones*"(Goodfellow et al. 2016, p. 8). The layered representation also raised interest in the reuse of trained networks across domains (Oquab et al. 2014). Hence, from the knowledge management perspective (cf. Section 3.1.3), trained networks themselves can become artefacts subject to knowledge transfer provided a shared representation format exists.

To conclude, sub-symbolic classification involves highly accurate techniques such as Support Vector Machines and Artificial Neural Networks which as one common denominator emphasise model performance in terms of predictive accuracy over model interpretability and the chance to explain classification decisions.

### 4.1.3 Integration of Base Learners through Meta Learning

The overview on machine learning methods for classification thus far enumerated base learning techniques. Building on this foundation, hybrid algorithms utilise ensembles of heterogeneous base learners with different inductive bias on the same classification task. Given a particular classification query, a meta learner then delegates the query to the ensemble and applies a judgement reconciliation strategy in order to determine the group decision.

Beyond concrete meta learning formalisms, the concept to integrate distributed knowledge from an ensemble is interesting both from the multiagent and the knowledge management perspective. However, while it is conceivable to implement meta learning in terms of a group of agents that assume knowledge management roles such as knowledge processing (for the base learners) and both mediation and integration (for the meta learning controller). The resulting agent federation

would have a holonic character.

In order to contribute to its team, each of the base learners would have to be online. Hence, while offering an approach to mitigate learning bias, meta learning does not serve to transfer knowledge permanently among base learners.

### 4.1.4 Extracting Symbolic from Black Box Representations

The ex-post deduction of arguably more comprehensible symbolic representations such as decision trees or classification rules (Freitas 2014) from trained sub-symbolic models is an active area of research. Guidotti et al. (2018) present a recent survey of methods for explaining different types of black-box models.

Barakat & Bradley (2007) discuss rule extraction from trained Support Vector Machines. Other contributions include (Zhang et al. 2005, Fung et al. 2005) and (Huysmans et al. 2008). Martens et al. (2009) use active learning (cf. Section 4.2.1) in order to extract rules from SVMs.

Contributions that cover specifically rule extraction from artificial neural networks have been surveyed by Augasta & Kathirvalavakumar (2012). Paradigmatically, Chorowski & Zurada (2011) extract rules from neural networks as decision diagrams. More recently, Chakraborty et al. (2018) have proposed Reverse Engineering Recursive Rule Extraction (RE-Re-RX) for symbolic rule extraction from neural network with mixed attributes. Due to the immense interest in Deep Learning and the associated deep neural networks, research in rule extraction algorithms for such networks has increased as well (Hailesilassie 2016, Zilke et al. 2016, González et al. 2017).

de Fortuny & Martens (2015) show that recent contributions increasingly focus on the *pedagogical* extraction of comprehensible knowledge from trained black box models. The distinctive feature of such approaches is their agnosticism with respect to the originally employed induction technique as they only employ the original model as oracle for instance labelling. Comprehensible classification models based for instance on propositional rules are consequently fitted to the responses of the oracle, often guided by an active learning process (Guidotti et al. 2018, p. 18).

The research touched above suggests the feasibility of establishing a syntactic interoperability between comprehensible symbolic classification models such as propositional rule bases, and probabilistic and sub-symbolic methods using model transformation. Even more importantly, results by de Fortuny & Martens (2015) furnish evidence that rule-based classifiers derived from black-box models can significantly outperform comparable classifiers that have been directly induced from training data.

## 4.2 Autonomous Control in Learning Processes

Base learners as discussed in Section 4.1 can be conceived as knowledge creation machines that transform a preset sample data input into the respective model representation. Supporting processes in knowledge acquisition that enable effective learning episodes as discussed before in Section 3.1.3 were not considered. For the application of trained learners, meta learning widened the research focus and showed that an integration of heterogeneous learners in ensembles using meta control to synthesise query results can be beneficial. Similarly, the idea of an autonomous control and shaping of learning processes beginning with proactive and targeted knowledge acquisition from human experts and fellow learners has been brought up. The corresponding efforts to introduce concepts of agency and agent-oriented knowledge management are discussed in the following sections.

### 4.2.1 Active Learning

Active learning (Cohn et al. 1994), also referred to as *query learning* or *optimal experimental design*, constitutes a subfield of machine learning whose key hypothesis can be concisely summarised as follows.

> If the learning scheme that is used to address a given learning task is *activated* such that it is allowed to autonomously and gradually acquire new learning data, it can perform better while training time is reduced as compared to a classical passive learning scheme.

Such a behaviour is desirable since for many real-world learning problems which are routinely addressed by a supervised learning system, said system typically needs to be trained on large amounts of labeled instances whose preparation may involve considerable costs in terms of required annotation time or resources that need to be spent, be they personnel or monetarial. A paradigmatic use case includes information extraction where learning requires labeled documents with detailed annotations. Pan & Yang (2010) report that locating entities and relations in simple newswire stories can take about half an hour or longer. The annotation of other texts, such as scientific texts, may in addition require considerable domain expertise of the annotator and can even prolong the process. Another use case involves document classification. Here, it is the annotation of sufficient amounts of documents that has proven to be time-consuming.

**Active Learning with Class Membership Queries** In his comprehensive survey on active learning, Settles (2009) distinguishes three fundamental scenarios for active learning processes.

a) *membership query synthesis*,
b) *stream-based selective sampling*, and finally
c) *pool-based sampling*.

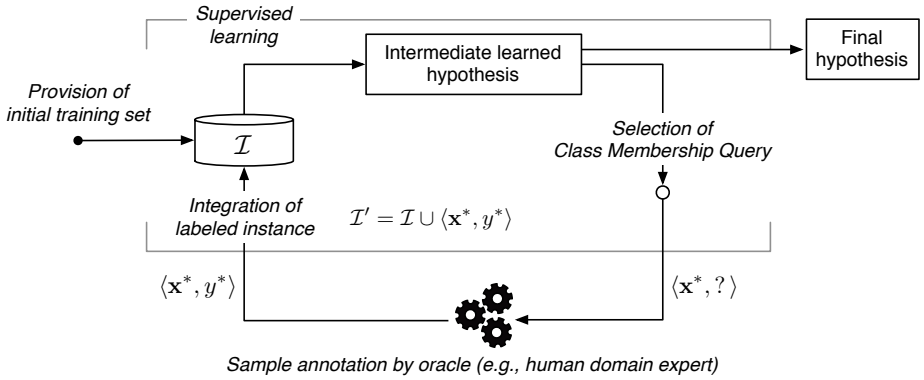



**Figure 4.3:** The basic active learning cycle (adapted from Settles 2009, p. 5).

All of these scenarios share a common learning cycle which is illustrated in Figure 4.3. It is assumed that at the beginning of the learning process only a rather limited set $I$ of labeled training instances $\langle \mathbf{x}, y \rangle$ are available to the machine learning system in order to form an initial hypothesis $h_0$, be it a classification or regression model. Active learning also presumes a data source from which additional samples, i.e., unlabelled training instances characterized only by their feature vector $\mathbf{x}$ can be drawn incrementally. Depending on the applied active learning scenario, this data source can take the form of a sample generator $\mathbf{g}(h_i)$ or a reservoir $\mathcal{U}$ of concrete instances which can be interfaced by the learner.

The learning cycle is initialised with the formation of an initial hypothesis $h_0$ which is learned using only the instances $\langle \mathbf{x}_i, y_i \rangle \in I$. Subsequently, the active learning process enters its first learning phase targeted at an improvement of the initial hypothesis. Based on guidance from a suitable *query selection strategy*, the learner typically designates a single unlabelled instance which is either drawn from $\mathcal{U}$ or compiled de novo using a generator function $\texttt{gen} : \mathcal{H} \mapsto \mathcal{X}$ where $\mathcal{H}$ denotes the hypothesis space of possible models and $\mathcal{X}$ the feature space. This instance is chosen such that the process of obtaining a label and subsequent addition to the training set $\mathcal{I}$ yields a significant improvement of the next-step hypothesis $h_1$ over the initial hypothesis $h_0$. The selected sample is presented as a query of the form $\langle \mathbf{x}^*, ? \rangle$ to a so-called *oracle* for annotation. Formally speaking, the oracle constitutes a function $\texttt{oracle} : \mathcal{X} \mapsto \mathcal{Y}$ which unequivocally assigns an element of the label space $\mathcal{Y}$ to an element in the feature space $\mathcal{X}$.

In typical applications of active learning schemes, the role of the oracle is filled by a human experimenter or domain-expert although alternatives such as the conduct of automated experiments do also exist in the literature (see, for instance, King et al. (2009)).

Following its annotation, the newly labeled instance is added to the training set, such that $I_1 \leftarrow I_{init} \cup \langle \mathbf{x}^*, \texttt{oracle}(\mathbf{x}^*) \rangle$. This extended training set is then used for the next pass of supervised learning that yields $h_1$. The learning cycle ends either due to compliance with a *stopping criterion* or limitations on run-time or annotation resources.

The scenarios for active learning differ in the way that new labeled samples are

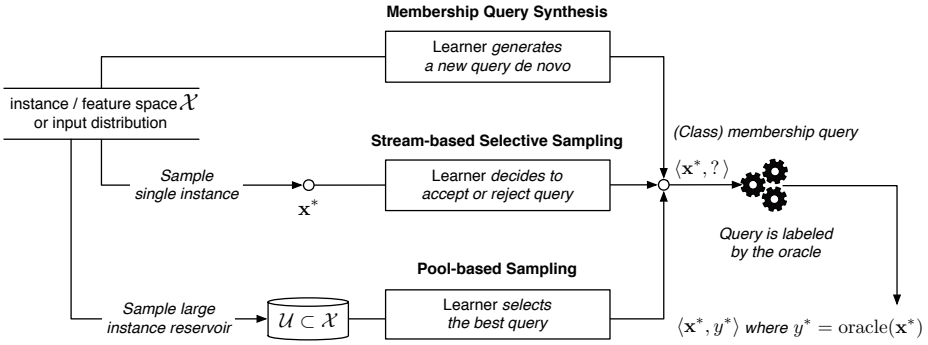



**Figure 4.4:** The main active learning scenarios (adapted from Settles 2009, p. 9).

drawn as queries to the oracle in each iteration. Their differences are illustrated in Figure 4.4.

In *membership query synthesis*, the learning system is allowed to *construct* a new unlabelled instance directly from the underlying instance space. Hence, it applies $\texttt{gen} : \mathcal{H} \mapsto \mathcal{X}$. While this course of action endows the learner with a considerable degree of freedom in its selection of 'helpful' samples, it can be demanding for the oracle as it may be asked to annotate synthetic instances which are not typically encountered in the real world. This means that $\mathcal{X}^{real} \subset \mathcal{X}$.

Both *stream-based selective sampling* and *pool-based sampling* by contrast constrain samples to be drawn from a real-world data reservoir $\mathcal{U}$. However, the sampling from $\mathcal{U}$ is realised differently in both scenarios. In stream-based selective sampling, the key assumption is that obtaining an unlabelled instance is either free or reasonably inexpensive. Thus, instances are sampled in a stream-based or sequential fashion and the learner subsequently renders the decision either to reject or accept the sample as next query to the oracle. The decision making thereby always pertains to the local assessment of a single sample. In pool-based sampling, by contrast, the learner is granted access to the whole reservoir $\mathcal{U}$ and is asked to performa a global assessment to determine the sample used in its next query.

Settles (2009) also provides a comprehensive introduction to query strategy frameworks which are used for the actual selection of queries. These include 1) *uncertainty sampling*, 2) *query-by-committee*, 3) *expected model change*, 4) *expected error reduction* 5) *variance reduction*, and finally 6) *density-weighted methods* .

Most work in active learning assumes that the employed *query unit* is of the same type as the target concept to be learned. Thus, if it is the learning goal to assign class labels $y \in \mathcal{Y}$ to instances characterized by their feature vector $\mathbf{x} \in \mathcal{X}$, the queries to the oracle take the form $\langle \mathbf{x}, ? \rangle$ and the oracle is to annotate the sample with the missing label. Researchers have also considered further types of queries and have thus extended the basic active learning framework in several ways adding new query-dimensions.

**Feature Queries for Feature Subset Selection** One alternative setting is the incorporation of queries that refer to features that characterise training

instances, in addition to conventional class membership queries.

Raghavan et al. (2006) have proposed one such approach which can incorporate feature feedback in classification problems. In their approach, a text classifier may interleave membership queries with questions about the *relevance* of features. A relevant feature is highly likely to help discriminate the positive class from the negative class. Highly-discriminating features in the feature vector characterising an instance are marked to reflect their relative importance. The accumulated feature annotations are then employed for a dimensionality-reduction of the original feature space $\mathcal{X}$ such that momentarily insignificant features can be masked, thereby optimising the input for the next supervised-learning iteration. Raghavan et al. propose a multi-dimensional active learning scheme where class membership and feature queries are interleaved. It should be noted that the feature queries differ from membership queries in that they are free from the context of a specific unlabelled sample $\mathbf{x}^*$ and refer to the full training set.

The authors found that human annotators performed well in answering feature queries in empirical user studies. In contrast to membership queries, feature queries do not establish any direct association between features and class membership.

**Feature Queries for Feature-Class Constraints** Several novel methods have been conceived for the incorporation of feature-based domain knowledge into supervised and semi-supervised learning. Human domain experts specify a set of constraints between features and class labels. A popular area of application for these methods addressed in the subsequently introduced research is text/document classification (See, for instance, (Druck et al. 2008, Sindhwani et al. 2009, Melville & Sindhwani 2009)).

Druck et al. (2008) argue that compared to traditional annotation in active learning using membership queries, asking domain experts to directly state feature-class constraints may be a more economical use of resources. The authors propose a method for training discriminative probabilistic models with labeled features and unlabelled instances. Unlike previous approaches that use labeled features to create labeled pseudo-instances, the labeled features are employed directly to constrain the model's predictions on unlabelled instances. In this context, Mann & McCallum (2008) have found that specifying many imprecise constraints is more effective than fewer more precise ones, suggesting that human-specified feature labels (however noisy) are useful if there are enough of them.

Sindhwani et al. (2009) have also explored interleaving class label queries for both instances and features. The authors refer to their approach as *active dual supervision*, which is used in a semi-supervised graphical model.

Similar to the feature queries used to guide feature subset selection introduced in the preceding paragraph, the feature-class constraints do not refer to specific labeled instances. They rather constitute pieces of explicitly stated domain expertise gathered, for instance, in a knowledge acquisition process. As a consequence, learning schemes which support this flavour of active learning go beyond standard supervised learning from feature vector / class tuples as they effectively need to incorporate additional background knowledge.

**Active Learning for Model Transduction** The fundamental setting of active learning has been modified to address the challenging task of a transduction of knowledge between different types of model representations. For instance, it might be desirable in some cases to initially induce a model using one type of model class, and subsequently try and transduce the gist of the former model's knowledge to a model of a different type and properties. For example, artificial neural networks have been shown to achieve better generalisation accuracy than decision trees for many applications. However, decision trees represent symbolic hypotheses of the learned concept, and they are therefore much more comprehensible to humans, who can inspect the logical rules and understand what the model has learned.

Craven & Shavlik (1996) have proposed the *Trees Parroting Networks* algorithm (TREPAN) to extract highly accurate decision trees from trained artificial neural networks (or similarly opaque model classes, such as ensembles), providing comprehensible, symbolic interpretations. Other researchers have adapted this idea to "compress" large, computationally expensive model classes (such as complex ensembles or structured-output models) into smaller, more efficient model classes (such as neural networks or simple linear classifiers).

These approaches can be conceived as active learning methods where the oracle is in fact another machine learning model (i.e., the one being parroted or compressed) rather than, say, a human domain expert. In both cases, the 'oracle model' can be trained using a small set of the available labeled data, and the *imitating model* is allowed to query the oracle model for labels of any unlabelled data that is available, or synthesise new instances de novo. These two approaches correspond to the pool-based and membership query scenarios for active learning, respectively.

### 4.2.2 Argument Based Machine Learning

The preceding paragraph gave a description of active learning approaches which featured queries that aimed at the iterative acquisition of background knowledge in the form of feature-class constraints. These constraints could be exploited in subsequent supervised learning passes. The notion to acquire background knowledge can also be applied to specific labeled training instances.

Možina et al. (2007, 2008, 2010) have proposed *Argument Based Machine Learning* (ABML). In contrast to the previously discussed active learning approaches that either exclusively or primarily build on the use of class membership queries as a means to pursue an iterative, goal-directed collection of additional training instances (as compared to random sampling), ABML builds solely on queries designed to acquire justifications, referred to as arguments, for the association of feature vectors $\mathbf{x}_i \in \mathcal{X}$ to given class labels $y_i \in \mathcal{Y}$.

Therefore, while the underlying learning cycle in ABML is in accordance with general active learning, the approach is special in that the absolute size of the training set is kept constant during iterative model improvement. Having learned an initial hypothesis $h_0$ using just the training set $I$ of labeled instances, the critical next process step in ABML consists of a self-assessment of the model

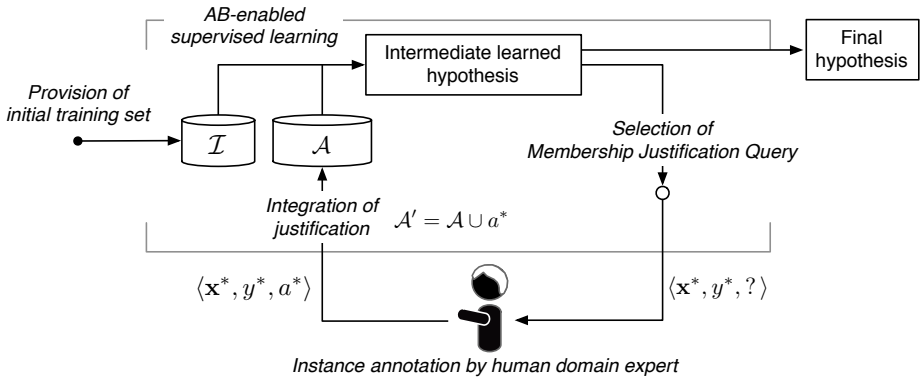



**Figure 4.5:** The active learning cycle for argument based machine learning.

performance that can be achieved in a repeated ten-fold cross-validation. This leads to the identification of so-called critical instances whose correct class is predicted incorrectly with a high frequency in the repeated cross-validation. Taking this as an indicator that these instances are probably not covered well by the current hypothesis, such that it would make sense to construct queries for instance-specific, and thus inherently local, expertise that explains the correct classification. Therefore, the queries posed to the oracle in ABML take the form $\langle \mathbf{x}^*, y^*, ? \rangle$ where the last spot in this triple stands for a class membership justification. Once such a query has been answered with a suitable annotation by the oracle, the justification is added to the growing pool of background knowledge on a small subset of all instances in $I$. For such background knowledge to be accommodated in subsequent passes of supervised learning, specifically extended versions standard machine learning algorithms are necessary.

In their experiments, Možina et al. have extended the CN2 rule-induction algorithm (see (Clark & Niblett 1989, Clark & Boswell 1991)) and called the new algorithm ABCN2 (Možina et al. 2007). Napierala & Stefanowski (2010) have extended an additional rule-induction algorithm, called MODLEM to create an ABML compliant version.

In either case, the employed learning algorithms ensure that uncovered generalisations induced from the training set account for the provided expert justifications. Algorithmically, the introduced expert knowledge constrains a combinatorial search of possible generalisations and directs it towards hypotheses that are comprehensible in the light of the specified knowledge. For the domain expert, it is not necessary to specify knowledge in terms of general rules which might be difficult to pin down. Explanations must cover only specific cases.

More recently, the principles of argumentation that enable argument based machine learning have also been applied in argumentation based multi-agent joint learning, a novel approach on ensemble learning by Xu et al. (2015) aimed at knowledge integration.

### 4.2.3 Semi-supervised Learning

Active learning and semi-supervised learning (Zhu 2005) constitute in some sense complementary techniques that can be applied by a learning system in scenarios which are characterized by a sparseness of labeled training data for pure inductive learning of classification or regression models.

As shown in the previous section, active learning presupposes that said sparseness of readily available training data can be mitigated by adoption of an active data acquisition strategy. The common underlying philosophy behind the various flavours of active learning is to enable the learner to conduct a self-assessment of its momentary model and identify areas or specific points in the feature space $\mathcal{X}$ where the uncertainty of the model, or the probability of incorrect predictions, are high. Then, additional training instances can be acquired from an oracle using, for instance, uncertainty sampling. In addition, or in the case of argument-based machine learning, additional background domain knowledge can be acquired and subsequently accommodated in further passes of supervised learning.

To sum up, active learning exploits known weaknesses of the current model. Semi-supervised learning by contrast seeks to exploit presumed strengths of existing models.

**Self-Training** This is exemplified in a basic semi-supervised learning technique by the name of *self-training* (Yarowsky 1995). In this approach, an initial hypothesis is learned using a limited training set $I$. As in basic active learning, it is then presumed that the learner can draw further unlabelled samples from some data source or generate them de novo. These samples are then again presented to an oracle for labelling and subsequent increase of $I$. The essential difference to active learning is, however, that the enlisted oracle is *not* provided by an external entity. Rather, the learner choses to simply employ its own imperfect model, yet choses to assign class labels only to drawn samples where the self-confidence in the prediction exceeds a certain threshold whose value may increase with the number of learning passes. Self-training can therefore be perceived as a kind of bootstrapping[1] technique.

**Co-Training** Co-training is a semi-supervised learning technique which involves an ensemble of base learners which are to engage in reciprocal training (Blum & Mitchell 1998). For this technique to be applicable several assumptions need to hold. First, it must be possible to partition the feature space $\mathcal{X}$ such that each partial feature space $\mathcal{X}' \subset \mathcal{X}$ is still sufficiently discriminative to train a good classifier. Also, these sets need to be conditionally independent given the target concept. Under these premises, the co-training process is then described as follows. In an initial phase, the involved learners each learn with their feature-constrained view on the common underlying training set $I$. Each learner then uses its initial model to predict class membership of further unlabelled samples that have been drawn from the feature space. Having singled out those samples

1 Bootstrapping here is not to be confused with *boosting* as ensemble learning method (see (Kononenko & Kukar 2007, p. 94)).

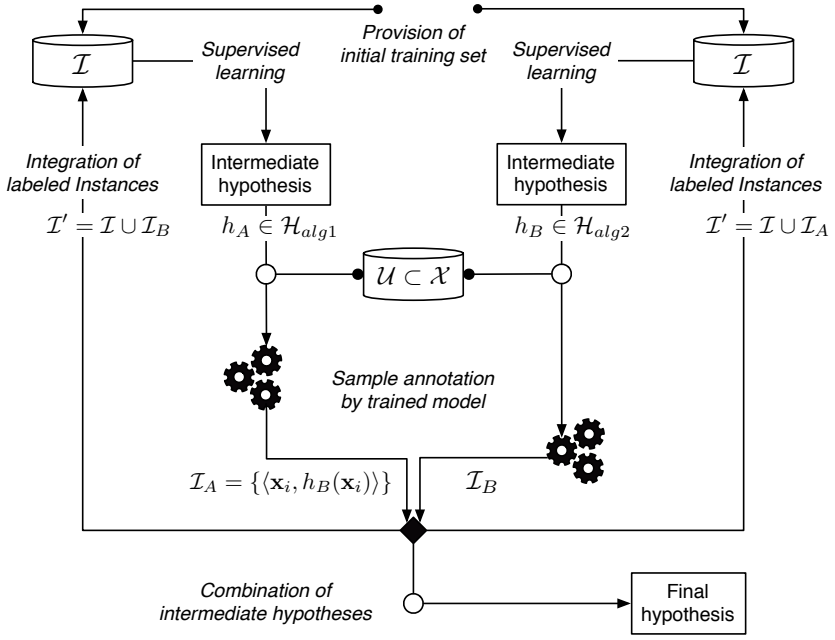



**Figure 4.6:** Illustration of a co-training process which involves two learners that apply different learning schemes and thus feature a different inductive bias, while both see the full feature vector of the initial training set. This kind of co-training conforms to Goldman & Zhou (2000).

whose classification can be computed with the highest confidence, each learner thus has produced a new additional set of self-labeled instances which can then be communicated to the peer learner as additional training material $I_i^j$ where $j$ identifies the '*teacher*'. The next supervised learning pass then uses $I_{i-1} \cup I_i^j$ as training set.

Co-training makes strong assumptions with respect to feature space partitioning. Hence, other researchers have investigated if these conditions can be relaxed. Goldman & Zhou (2000) propose co-training with an ID3 decision tree learner and a so-called HOODG learner[2]. However, both learners are granted access to the complete feature set, and essentially use one learner's high confidence data points, identified with a set of statistical tests, to teach the other learner and vice versa. Chawla & Karakoulas (2005) perform empirical studies on this version of co-training using Naïve Bayes and a C4.5 rule induction learner. They compared the results against several other methods, in particular for the case where labeled and unlabeled data do not follow the same distribution. Zhou & Goldman (2004) propose a single-view multiple-learner Democratic Co-learning algorithm. An ensemble of learners with different inductive bias are trained separately on the complete feature set of the labeled data. They then make predictions on the unlabeled data. If a majority of learners confidently agree on the class of an unlabeled sample, that classification is used as the label. The thus labeled instances are then added to the training data. All learners are retrained on the updated training set. The final prediction is made with a variant of a weighted majority vote among all the learners. Zhou & Li (2005) propose

2 Instead of decision trees, HOODG induces an *oblivious read-once decision graph (see Kohavi (1994))*

'*tri-training*' which uses three learners. If two of them agree on the classification of an unlabeled point, the classification is used to teach the third classifier.

By comparison with the descriptions in Section 4.2.1, one can conclude that active learning and semi-supervised learning attack the same problem from opposite directions. While semi-supervised methods exploit what the learner thinks it knows about the unlabeled data, active methods try and explore the unknown aspects. This has led researchers to consider hybrid approaches for semi-supervised active learning (see Tomanek & Hahn (2009)).

## 4.3 Multiagent Learning

The previous section has shown that the machine learning community has adopted the concept of machine learning systems as systems with

a) considerable agent characteristics such as a clear learning agenda,
b) goal-directed pro-active behaviour in pursuit of this agenda,
c) the capability to interact with the environment through perception and actuating elements,
d) and finally a degree of social ability expressed in the ability to entertain structured interaction with human experts.

The shift from the focus on single intelligent agents to groups of agents influences the circumstances, opportunities and, potentially, the motivation for agent-based learning.

The branch of agent-based learning covered in Section 2.3 is denoted as *single-agent learning*. Figure 2.4 on page 20 illustrated the assumed setting for single-agent learning, which demands only a single learner, embedded within a potentially complex environment. No further assumptions are made whether or not this environment hosts further intelligent agents. As far as learning is concerned, fellow agents are treated simply as aspects of the environment. The protagonist in single-agent learning may or may not be aware of their presence. In any case, it is not reflected in the pursued learning goals nor the process and means to reach these goals.

The overarching theme is consistently for the agent to retain or become more competitive in its primary domain tasks. In essence, single-agent learning is a means to enhance individual competencies, dexterity and knowledge.

However, if one adopts a more differentiated view of the environment that embeds the learner, a common observation will be that the environment often hosts multiple protagonists that exert single-agent learning concurrently or that need to coordinate their action to achieve a shared goal. The transition from single- to multi-agent learning is therefore fluent. Panait and Luke provide an intentionally coarse characterisation for multi-agent learning which tries not to be specific to any research community contributing to the field. According to them "*[multi-agent learning] is the application of machine learning to problems involving multiple agents.*". They also note that "*multi-agent learning may involve multiple learners, each learning and adapting in the context of others*" (Panait &

Luke 2005, p. 389).

Hence, concurrently exerted single-agent learning within the same environment turns into multi-agent learning when the learning explicitly considers other relevant protagonists.

Weiß & Dillenbourg (1999, Chap. 4) have proposed a further distinction of concurrent learning, focussing on the way learning tasks are distributed in a multiagent system and also on the interplay of learning activities: 1) *Multiplication mechanisms* (Multiplied Learning), 2) *Division mechanisms* (Divided Learning), and 3) *Interaction mechanisms* (Interacting Learning).

### 4.3.1 Multiplied Learning

Multiplied learning is the most rudimentary form of concurrent learning, in the sense that it is achieved simply by providing multiple protagonists within a multiagent system with a learning element as introduced for single-agent learning in Section 2.3. That is, each learner acts independently from peers such that eventual interactions among the agents have no further influence on their respective learning processes. "*Not the agents, but their learning processes are, so to speak, isolated of each other. [...] each individual learner typically pursues its own learning goal without explicitly taking care of other agents' learning goals and without being guided by the wish or intention to support the others in achieving their goals*" (Weiß & Dillenbourg 1999, Chap. 4, p. 6)

The authors note that the learning goals may of course mutually supplement each other. However, this is conceived as a welcome emergent side-effect rather than a defining characteristic for multiplied learning. Graça and Gasper elaborate that since the agents do not share results, their learning efforts are multiplied by every individual involved. Hence "*multiplied learning can be a solution when agents cannot share information (privacy, hostility) or when communication cannot be supported (communication to expensive, lack of resources).*" (Graça & Gaspar 2004, p. 95).

This position is assured by Stone & Veloso (2000). At the group level, positive effects of multiplied learning include redundancy in case of multiple identical agents. Since expertise through learning is build up at different sites, the outage of specific actors can be compensated. Also, if the learners are deliberately designed to exert a range of competing learning schemes on the same learning problem or can base learning on differently biased input, the multiplication can provide the foundation for better model performance through additional adoption of the meta-learning principles.

### 4.3.2 Divided Learning

Divided learning constitutes the next stage of concurrent learning whereby a single complex learning task is divided among several agents. According to Weiß & Dillenbourg (1999), the division can be applied according to functional aspects

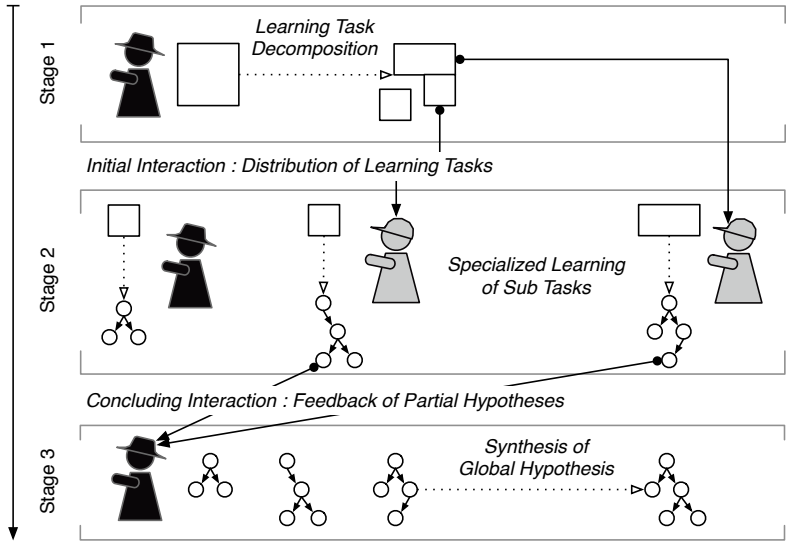



**Figure 4.7:** Schematic three-tiered structure of an divided learning process with multiple learning protagonists.

and/or the characteristics of the data, which is processed to attain the learning goals. A paradigmatic example for a functional devision can be illustrated by the global task to learn an effective team play in robotic soccer. Each participating agent in the team could focus on quite specific roles and their interplay with related roles. As an example for a data-driven division one could think of the task to classify land cover types from geographically distributed data as in (Blackard et al. 1999) or the classification of brain tumour types from clinical data raised by different medical facilities (González-Vélez et al. 2009). In these cases, different agents may be responsible to process data from different sources.

The agents involved in divided learning have a joint overall learning goal. The division of the learning task is typically conceived at design-time of the multi-agent system. Hence, it is not incorporated as a proper element of the applied learning scheme itself. "*Interaction [among the agents] is required for putting together the results achieved by the different agents, but [. . .] this interaction does only concern the input and output of the agents' learning activities. Moreover, the interaction does not emerge in the course of learning but is determined a priori.*" Weiß & Dillenbourg (1999, Chap.4, p. 8) Thus, agents in multiplied learning are equipped with the ability to interact through communication. However, the degree of freedom in the interaction is restricted. For instance, it is known a priori what information needs to be exchanged at which point in the team learning process. And if information or knowledge is shared, its further use is also preset.

Since each individual in divided learning effectively acts as a specialist, this kind of concurrent learning effectively follows a divide-and-conquer philosophy. Thus, the approach is somewhat related to distributed problem solving. Dillenbourg and Weiß note that multiple benefits arise from the division of labor. First, the functional decomposition of the learning task may facilitate the learning schemes used to address the individual sub-problems. Also, a speedup in learning may be achieved in scenarios where the time gained by the introduction of parallelism is

not weighted out by the necessary overhead for coordination and the consolidation of learning results.

In comparison with multiplied learning, divided learning trades in much of the formers redundancy to gain efficiency.

Ontañón & Plaza (2010*c*,*b*,*d*) investigate *empirical argumentation* wherein *inductive concept learning* is combined with *defeasible argumentation* (Bench-Capon & Dunne 2007) in *multi-agent inductive learning*. Groups of agents are faced with the task of finding a mutually accepted generalisation of shared concepts (*concept convergence* (Ontañón & Plaza 2010*b*)). Each agent thereby initially learns its generalisation, which can be represented as a rule set, based on its restricted empirical data. The agents subsequently engage in an argumentation process. They present constituents of their respective models, i.e., rules, as arguments to their peers. These can then either be accepted or challenged with counter-arguments in the form of more specific yet disagreeing rules (Ontañón & Plaza 2010*c*,*b*,*d*) or specific counter-examples (Ontañón & Plaza 2009, Ontañón et al. 2010, Ontañón & Plaza 2010*a*). Over the course of this process, initial generalisations can become defeated. However, the agent that originally suggested them takes the chance to revise its model taking into account the additional information contributed in the argumentation process. The approach is thus far devised for concept learning only (Ontañón et al. 2010). The intensional generalisations learned by agents are restricted to rule and decision tree base learners (Ontañón & Plaza 2009). The authors tacitly assume that suitable collaboration partners are known apriori.

Nielsen (2006) and Nielsen & Parsons (2007) advocate the adoption of defeasible argumentation in multi-agent systems where each individual agent is equipped with a local Bayesian network and the goal is to agree on a global 'consensus' network.

Ontañón & Plaza (2007*a*,*b*) investigate collaboration between *case-based reasoning* agents. In particular, the authors propose a proactive communication process for CBR agents to send problems to congeneric agents and consequently acquire respective justified answers. This process is engaged to acquire information which constitutes the basis for later informed case bartering which allows the agents to actively replenish their case bases with additional 'useful' empirical data from peers.

### 4.3.3 Interacting Learning

Interacting learning designates learning schemes whose successful application relies on learning-centred interaction during the learning process. While some restricted interaction may occur in the previously examined forms of concurrent learning, these interactions as a general rule conform to simple, predetermined protocols. Typically, they are confined to the delegation of local learning tasks, distribution of input data and combination of local learning results, or a subset of these. Weiß and Dillenbourg pose that "*in the case of interactive learning the interaction is a more dynamic activity that concerns the intermediate steps of the learning process. [...] interaction does not just serve the purpose of data exchange, but typically is in the spirit of 'cooperative, negotiated search for a*

*solution of the learning task'*" (Weiß & Dillenbourg 1999, Chap.4, p. 9).

An agent involved in interactive learning does not so much act as a '*generalist*' as in *multiplied learning* or '*specialist*' as in *divided learning*. In fact, the agent assumes the role of a '*regulator*'. This means, that it has full autonomy to continuously shape its own *learning path*. In the process, the agent also assumes an '*integrator*' role if need arises. In this role context, it synthesises the different perspectives and biases of stakeholders contributing to its own learning process. Graça and Gaspar state that two core principles of interacting learning are " *1) to let the succession of environment events and social interactions guide the path of the learning process; 2) to use the potentialities of multi-agent systems on behalf of the learning process.*" (Graça & Gaspar 2004, p. 96). Learning hence ceases to be a process which is isolated from other social processes.

Two strands of interacting learning can be distinguished conceptually. For the purpose of this work, these will be denoted as '*conversational*' techniques on the one hand, and '*activity-oriented*' techniques on the other hand.

**Activity-oriented Interacting Learning** In *activity-oriented* techniques, interaction is characterised to a large extent by direct *demonstration* and *imitation* of decision making, means-end-reasoning, or motion sequences which are subject to learning activity by an inexperienced agent. These techniques are geared towards learning a know-how for which it is easier just to show and take in desired behaviour than to compile and subsequently assimilate an adequate conceptualisation. According research, for instance in developmental psychology (Tomasello 1999, Tomasello et al. 1993) and sub-fields of biology (Nehaniv & Dauntenham 2007), has been inspired from learning in non-artificial multiagent systems, either humans or groups of social animals.

Specifically in biology, learning in multiagent systems has been studied under the heading of *social learning*. This form of learning actually constitutes a general learning concept which can be implemented with a variety of mechanisms such as following or stimulus enhancement, contagious behaviour, emulation or imitation which demand a growing set of cognitive skills from the learner to be applied successfully. For further information on the aforementioned mechanisms, see Noble & Todd (2002) and Tomasello (1999).

Noble & Franks (2003) observe in reviewing the literature on social learning in biology and developmental psychology, that authors often concentrate on a detailed analysis of the contexts in which the various forms of such learning take place. Nevertheless, potential benefits of social learning have recently attracted considerable interest in the robotics community as a way to interactively learn, rather than having to tediously engineer, behaviour or locomotion patterns (Nehaniv & Dauntenham 2007). Social learning has also been successfully applied to increase the performance of reinforcement learning (Noble & Franks 2003, Shon et al. 2007).

The role schema which forms the basis of activity-oriented techniques to interacting learning is similar to the conversational case in its distinction of expert and relative novice. However, the expert can be better thought of as assuming a kind of *trainer* role. The novice assumes a *trainee* role.

While the conversational techniques to interactive learning by trend originate

from research focussing on machine learning and its extension to multi-agent systems, the activity-oriented approaches are rather rooted in distributed artificial intelligence and robotics, especially in physical domains that abet the direct observation of the actions of the activity of other agents.

Sen & Kar (2002) investigate the research problem of how an adaptive agent can transfer its learned knowledge, i.e., a concept description, to train another agent with a potentially different internal knowledge representation. Within the *Agent Teaching Agent* (ATA) framework, the authors propose an algorithm that can be used by the trainer (an instance-based concept learner) to iteratively select training examples for the trainee (a decision tree learner). Chakraborty & Sen (2006) employ the ATA framework to transfer coordination knowledge between agents in a predator-prey pursuit environment. This work is related to a large body of research on collaboration in reinforcement learning such as (Nunes & Oliveira 2003, 2004).

**Conversational Interacting Learning** Although boundaries are blurred, the interaction in conversational techniques is typically characterized as a dialogue on relevant learning topics. As to addressed learning targets, these techniques are geared towards learning of *know-what*, such as decision support models. For such learning tasks, it is feasible to acquire well-structured learning advice in a conversation. This feasibility has been shown successfully in argument based machine learning (cf. Section 2.3).

The role schema which forms the basis of conversational techniques is analogous to that in active learning. On the one hand, there is a (semi-)expert for a particular domain context which assumes the role of an *advisor*. On the other hand, there is the active learner which is not as experienced and assumes an *advisee* role. Advisee and, potentially multiple, advisors then seek to achieve learning progress in a learning-centred dialogue.

The interactive circulation of knowledge that is related to specific learning tasks lies at the core of conversational interactive multiagent learning. It allows the learners to share and complement their individual experience. Against the background of the knowledge pyramid introduced in Section 3.1.2, it is necessary to define more precisely what the term 'circulation of knowledge' actually boils down to. In this regard, Graça & Gaspar (2004) pose that agents can share data in the form of samples from their respective data bases or compiled knowledge in the form of learning results, such as a classification model. In addition, the agents may also share specific units of information and knowledge that enable a richer interaction during the learning process. As examples for such units of information, the authors mention amongst others a problem description, a solution (for instance, as response to a previously stated problem), a justification that supports a particular solution, or an evaluation that provides as quality assessment for a solution. A typical characteristic of interactive multiagent learning is that the units of information are compiled by the protagonists of the learning basis as need arises. They reflect the current level of experience and learning progress of the agents.

The following enumeration lists examples for message contents that can be used to structure the circulation of knowledge among agents at the intermediate stages

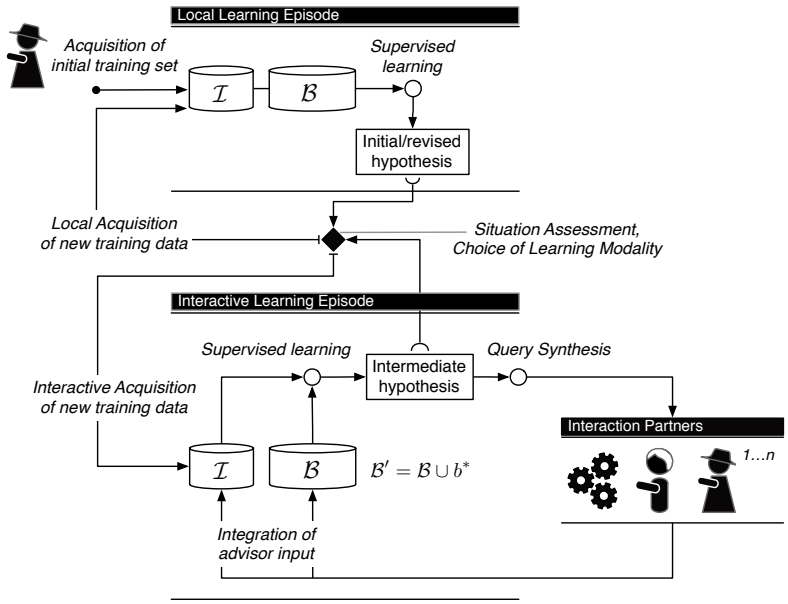



**Figure 4.8:** Schematic structure of an interacting learning process. Although similarities to active learning (cf. Figure 4.3) and argument-based machine learning (cf. Figure 4.5) exist, interacting partners in interacting learning are typically, but not exclusively, one or more other agents in the multiagent system.

of the learning process:

**Suggestion** Possible solution for a learning (sub-)problem.

**Opinion** Quality assessment of a proposed solution for a learning problem.

**Help Request** Request for learning advice from another agent that includes a problem description, i.e., the learning objectives for the interaction. The request may also include a suggestion and its justification.

**Advice** Response to a help request that may include a suggestion and its justification.

The examples above are concrete forms of knowledge sharing where, when in need, agents enlist the help of cooperating colleagues. In order to fulfil the prerequisites for interacting multiagent learning, it may be necessary to circulate further meta-information about the availability and capabilities of learners in the multiagent system. Thus a foundation is provided for any learner to reason about learning sub-tasks that can be delegated and the selection of peers to contribute to an interactive learning process.

Costantini & Tocchio (2005) have proposed an approach for learning by knowledge exchange in logical agents. The authors concentrate on the exchange of agent beliefs and rules encoding action recipes. They contemplate the role of trust in exchanging knowledge and propose strategies for graded operationalisation of acquired knowledge. Jakob et al. (2008) implement adaptability in multiagent systems by means of collaborative logic-based learning, focussing on communication strategies for acquired knowledge.

Tožička et al. investigate a market-based approach to collaborative learning in multiagent systems (Tožička et al. 2006, Jakob et al. 2008). Therein, *Inductive Logic Programming*, which represents both training data and learned generalisations in the Horn fragment of first order logic, is used for individual learning of prediction models for raid avoidance in an adversarial logistic environment. The authors propose a knowledge trading approach governed by a variant of the Contract-Net Protocol (Tožička et al. 2006). Agents are classified with regard to learning ability, readiness to assume a knowledge provider role, and trading conduct. Investigated are the effects of bartering both models and empirical data in different multiagent system compositions with respect to prediction quality, communication and computation load.

## 4.4 Chapter Summary

Following an initial familiarisation with the machine learning landscape in Section 4.1, the chapter focussed on aspects of autonomous control in learning processes in Section 4.2. It has been established that machine learning algorithms can take an active role in a particular learning task. Whether in active learning (cf. Section 4.2.1), argument-based machine learning (cf. Section 4.2.2) or semi-supervised learning (cf. Section 4.2.3), there is a trend in the development of machine learning schemes towards deliberation and an active exploration of the world. Even more interestingly from the learning agent perspective in Section 2.3 and also the agent-oriented knowledge management perspective in Section 3.4 is the fact that approaches like active learning are deeply built on continued knowledge acquisition through interaction with their environment using communication or goal-directed actions on the one hand, and an interleaving of learning and interaction on the other hand.

After the survey on activated learning schemes with immediate relevance for the implementation of agent-oriented knowledge management that have been proposed by the machine learning community, Section 4.3 has approached activated learning from the multiagent system perspective via a categorisation of multiagent learning.

Distributed learning as examined in Section 4.3.2 adheres to a divide-and-conquer strategy for the efficient solution of learning tasks pertaining to a fixed group of agents. Hence, the decomposition of the global learning task into independent chunks that could be handled by specialised learners, the distributed compilation of partial solutions, and their eventual synthesis into the desired global solution is the primary objective. A distinctive feature of interacting learning discussed in Section 4.3.3, by contrast, is the possibility for the formation of dynamic knowledge networks by agents whose individual learning targets become interlaced for a limited time. Interacting learning can accommodate a less pre-structured learning environment due to the fact that the acquisition of suitable interaction partners is incorporated as an important aspect of the learning process. Weiß & Dillenbourg (1999) and Graça & Gaspar (2004) suggest that interacting learning is a suitable mechanism to address not only team-oriented learning targets which are specific to multiagent systems but also for learning tasks that rather concern the individual

agent.

Interacting learning is related to the active learning approaches in Section 4.2.1 insofar as interaction in both cases is an integral element over the whole course of the learning process. Furthermore, the flow of interaction is not simply conceived at design time of the learner, be it a dedicated machine learning system or an agent, but is pro-actively managed and developed according to the requisites defined by the pursued learning goal and the environment.

The concerted delegation of specific learning sub-problems and the explicit acquisition of learning advice are consistently employed to further the learning process. The difference though between multiplied learning and active learning is reflected in the respective learning setting. In active learning, the learner operates in an environment with capable human domain experts. These are typically not ostensibly engaged in learning processes of their own but are rather seen as sources of knowledge that can tapped by the learner. As shown in Figure 4.8 above, interacting learning substitutes these human supporters by other learning agents. The idea thereby is to best exploit both the distributed learning capacities as well as the distributed acquired competencies with regard to common learning targets.

Chapter 5

# Interactive Multiagent Adaptation of Classification Models

Based on the groundwork that has been established in the preceding review chapters, this chapter consequently presents a conceptual foundation of a flexible methodology for interactive adaptation of individual classification models as decision support enablers in multiagent systems. The concept blends aspects from multiagent systems, active machine learning, and agent-oriented knowledge management. It spans the necessary constituents that enable a self-contained subsystem that can be embedded into the knowledge management modules of knowledge-based, learning agents. The introduction of this methodology follows a top-down approach.

To begin with, Section 5.1 analyses scenarios and prerequisites for interactive multiagent adaptation. In particular, it identifies potential participants in model adaptation with their respective incentives.

Section 5.2 subsequently works out essential knowledge management functions that enable interactive model adaptation in knowledge networks emerging within multiagent systems. These knowledge management functions are associated with roles to be embodied by agents within knowledge networks. These identified roles can be understood as a first coarse structuring of the adaptation problem. They comprise a description of capabilities. In the subsequent section these roles are fleshed out.

Section 5.3 puts the advisee role into focus. Classification model adaptation that is sought by agents assuming this role is modelled formally in terms of an online search problem. It interleaves the acquisition of learning advice and its exploitation through model revision. The choice of appropriate advisors for a concrete adaptation intent is described as an integrated part of the control process for adaptation episodes.

Having outlined the application flow for the model adaptation process, Section 5.4 provides details with respect to the essential building blocks for the implementation of the adaptation concept. These comprise the nomination of appropriate advisory topics by the advisee throughout the adaptation process (Section 5.4.1).

Next, the structuring of the advice acquisition protocol to address the aforementioned learning problems (Section 5.4.2). A formal definition for learning advice is presented. It is discussed, how actual advice is procured by agents in the advisor role (Sections 5.4.3 and 5.4.4). The consolidation of advice on a certain learning problem, which has been acquired from multiple sources, is introduced (Section 5.4.5). In the process, the possibility for advice generalisation is also covered. The utilisation of processed learning advice to obtain new classification models with improved performance is discussed (Section 5.4.6).

The chapter closes with a summary of contributions.

## 5.1 Requirements and Basic Setting

As point of origin for the development of a methodology enabling knowledge-based learning agents to utilise peer support in their pursuit of individual learning of classification models, this thesis presumes a multiagent system that hosts a multitude of knowledge-based software agents. Knowledge-based, in this context, is to imply that the decision making in the execution of domain-specific agent roles benefits from individual classification models for situation assessment and prediction.

Since the term decision support model comprises a considerable bandwidth of model families and variations, it is qualified for the remainder of this chapter with the help of the taxonomy of machine learning tasks illustrated in Figure 4.1 on page 62 in Section 4.1. The focus in this thesis is put on models that are for the most part induced with supervised machine learning schemes, specifically models that are used for classification tasks. Therefore, unless otherwise specified, the term model is used hereafter interchangeably with decision support model.

It is assumed that for a sufficient fraction of knowledge-based agents in the agent community, their productively employed models are not unique with regard to their model domain. This term refers to a) the underlying feature set for the characterisation of instances and b) the target concept with its class partitioning. Following this non-exclusivity assumption, models with accordance on the domain level are used by multiple agents. It is hence feasible, conceptually, to cluster active agents in a multiagent system into potentially overlapping groups according to the domains of their deployed models. Agents that fall into such a group share a common interest in capable classification models within a specific domain.

An additional assumption pertains to the origin of employed models. While in principle, models could be handcrafted in a knowledge engineering process or learned offline from historical data, it is presupposed here that at least a proper subset of users of a model characterized by its task-specific domain is endowed with a learning element (cf. Section 2.3). Hence, these learning agents have the capability to assume an active role in the preparation, fostering, and continued refinement of their respective individual model instances. They pursue agent-oriented knowledge management functions, amongst them data acquisition for learning tasks, data (pre-)processing, the application of learning schemes to induce a conceptualisation and hence new knowledge from the interpreted

input data, and finally the assessment and operationalisation of that knowledge (cf. Figure 2.6). Revisiting the agent grouping brought forward in the preceding paragraph, the constitution of an additional aspect of these groups becomes evident which applies specifically for agents with the ability to engage in agent-oriented knowledge management.

No longer is the common interest in a model with a specific domain only related to the application of one's own model instance in the decision-making processes within an application context. The joint interest in fact also extends to aspects of model preparation and maintenance. From a knowledge management perspective, knowledge about the existence of outlined groups would create opportunities for cooperative knowledge management activities. These can be understood as a complement to the aforementioned self-sufficient activities.

The baseline for this line of reasoning is founded in the observation that each instance of a model with the same model domain, used by a knowledge-based agent within an interest group, is typically built from a different pool of original data and domain background knowledge. While handcrafted models constitute a direct formalisation and codification of tacit domain expertise, the preferred approach in many application domains is the training of classifiers predominantly based on raw data. The training data which is used by different learning agents in general constitutes a restricted and potentially biased sample from an application domain. In addition, heterogeneous learning schemes implicate differences with regard to the respective inductive bias.

As a consequence, the models which are employed by agents within a common group are expected to reflect different approximations or conceptualisations of the same concealed patterns/relations. In this situation lies an opportunity for those agents whose models can be classified as low-performers in comparison to those of other group members. Assume the possibility to partition an interest group such that a subset of agents with a high model performance can be identified. These relevant agents are referred to as *semi-experts* for a particular model domain. The term semi-expert has been chosen intentionally to distinguish such agents qualitatively according to their model performance and place them between low-performing agents and human domain experts. It can now be argued that the distinction of low-performers and semi-experts within a group has denominated the potential stakeholders in a task-specific *learning setting*. One the one hand, there are actors with a motivation to improve their momentary productive model in order to mitigate an existing performance gap towards 'expert models'. On the other hand, there are actors whose models constitute knowledge assets for which a market exists. In a cooperative environment, this could create incentives for the provision of novel knowledge-based services. These would be designed to further learning and model adaptation activities from low-performers.

To conclude, a potential for domain-specific learning settings in knowledge-based multiagent systems has been identified, including stakeholders with a motivation to draw on services to further their individual learning activities. Their role within the learning setting is that of *knowledge consumers*. The complimentary role of stakeholders which offer learning support is consequently that of matching *knowledge providers*. The relation between both roles, to be elaborated in Section 5.2, and indeed the complete learning setting is equivalent to the situation which has been outlined in Section 4.2.1 on active learning and

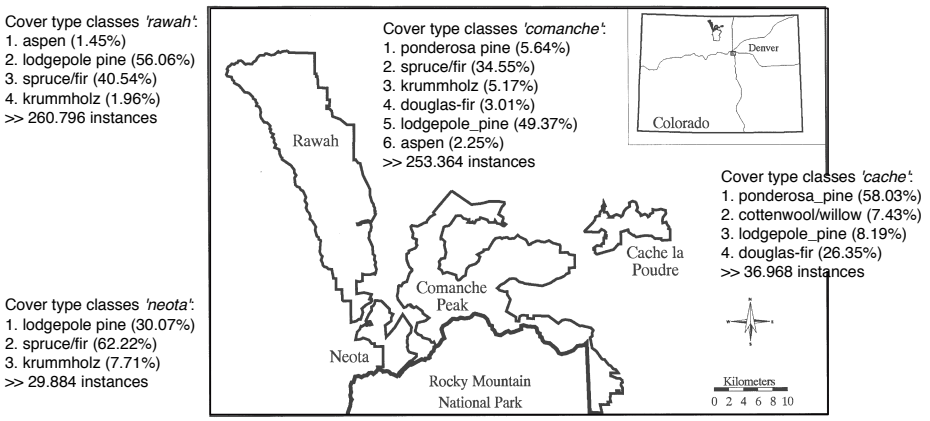



**Figure 5.1:** Overview of the forest cover type data set used by Blackard and Dean (Blackard et al. 1999).

in Section 4.2.2 on argument based machine learning. A distinctive characteristic of the situation here is the potential availability of multiple interaction partners. Hence, agents that assume a *knowledge consumer* role are not confined to a single source to acquire additional knowledge.

The outlined scenario is of particular interest for persistent, open multiagent systems. Persistent here means that these multiagent systems are active for extended periods of time. They are open in the sense that they are subject to dynamic change with respect to the composition of the active agent population.

The intent in a strategic proactive management of an agent's knowledge assets – i.e., its productive models – as discussed in Section 2.3.1 warrants the competency of the examined agents to render informed knowledge-based decisions consistently without the coercively necessary consultation with further agents at decision time. By contrast, the agents get the chance to treat interactive model adaptation as an investment which not only improves model performance immediately but also permits to retain such added value in the long-term, even in situations when the original contributors of learning advice are no longer active in the multiagent system.

The presented scenario fits the methodology of interacting multiagent learning introduced in the previous chapter insofar as knowledge management activities that agents perform to learn or adapt their own models become intertwined with and profoundly shaped by knowledge acquisition. It can be developed on a conceptual level with the framework of role-based, distributed knowledge management. In terms of a concrete embodiment, it can draw from the argument based machine learning when it comes to the utilisation of learning advice. Finally, agent coordination mechanisms can be used to make the step from the conceptual interest group to instantiated knowledge networks.

Up to this point, the general scenario for interactive adaptation of individual decision support models in multiagent systems has been introduced. This scenario is a recurring challenge in multiagent systems.

**Example 5.1.** *The following example considers an application from the field of forest science. The learning task which is addressed pertains to the classification of*

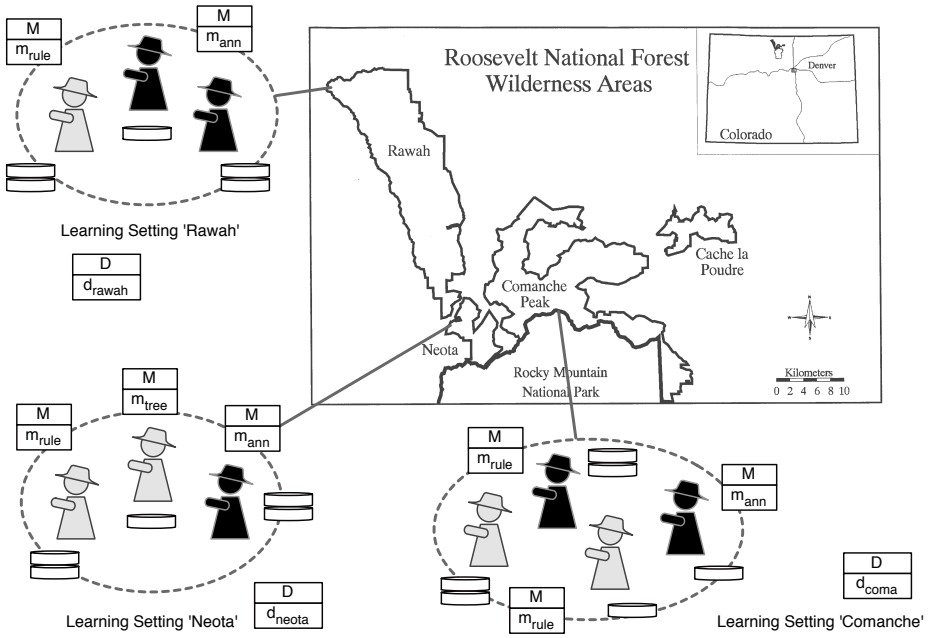



**Figure 5.2:** Exemplary mapping from classification tasks for particular wilderness areas, signified by their model descriptor $d_i$, to task specific learning settings and associated agents. While it is presumed that each partaking agent maintains its own model, the model types $m_j$ may be heterogeneous within the multiagent society.

*mutually exclusive types of forest covers in several wilderness areas of the Roosevelt National Forest located in Denver, Colorado. This classification problem and the underlying real-world data set, which is based on digital spatial data provided by the US Geological Survey and the US Forest Service has been first described in Blackard et al. (1999).*

*Consider that a forest agency would like to build an information system for its staff which is backed by an expert system which determines the forest cover types for requested patches of any part of the administrated wilderness areas. A scenario can be construed where the presented classification task is decomposed such that each wilderness area is handled by a dedicated software agent with learning capabilities. Now assume that the forest agency would at some point like to introduce an improved learning agent. Then, there is clearly a motivation to benefit from the existing knowledge of the predecessor. Also, another agency may develop a similar agent-backed expert system with their own team of agents handling the classification task.*

*If these organisations run their own data acquisition processes, even agents which are assigned to the same wilderness area learn models on different data samples from the target domain. In this context, one can envision that these different data sources and supposedly different applied learning schemes and/or knowledge representations for the classification models with a common domain, result in variable model performance and hence quality of the agent-procured decision support services.*

*Spinning this use case further beyond the land cover problem, the same agents are also facing classification problems with a forestry background, such as the*

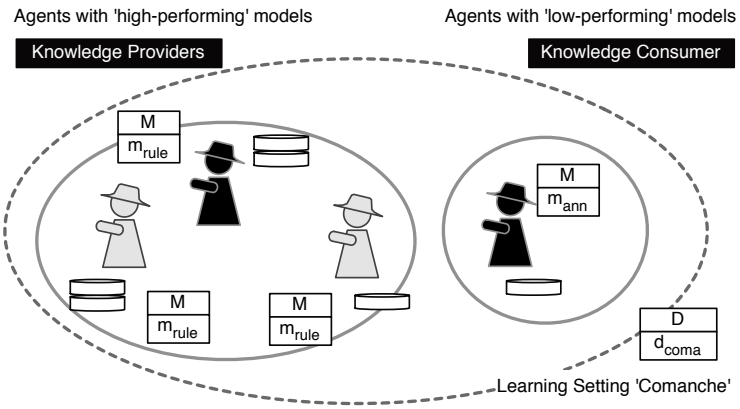



**Figure 5.3:** Partition of the agents within the same learning setting into high- and low-performers with respect to model performance and potential knowledge management roles.

*designation of tree and bush types based on features extracted from pictures of specimens. This is an example for different learning settings that emerge within a single multiagent system. Figure 5.2 illustrates the mapping into a multiagent scenario.*

Similar application scenarios have been described, for instance, in the health care area. González-Vélez et al. (2009) present an agent-based distributed decision-support system for the diagnosis and prognosis of certain brain tumours. They use multiagent technology to interconnect different brain tumour centres which face the same type of classification problem and each possess an individual case repository as input data. Warden et al. (2012) report on a scenario from the field of autonomous logistics where agents that represent individual means of transport of a freight forwarder individually learn classification models for environmental parameters such as traffic flow or weather conditions, based on their own observations. These scenarios exhibit an inherent distribution of classification models within agents, intersecting application areas, and a cooperative setting which abets the collaboration in knowledge management tasks.

## 5.2 Knowledge Management Roles for Classification Model Adaptation

The preceding section sketched an exemplary scenario for multiagent systems that can host domain or task-specific learning settings for loosely coupled groups of knowledge-based, learning agents. It has been argued that these learning settings provide an opportunity for agents partaking in these groups to complement their existing knowledge management skills, specifically data-driven model acquisition. The base line of the argumentation was that for each learning setting, it is feasible to assume a heterogeneous state of knowledge with regard to the respective model domain.

As a consequence, agents can be subdivided into different sub-groups according to

their model performance. Agents in the group of low-performers can be imputed a motivation to harness support of knowledgeable peers to attain a long-term improvement of their productive model. Hence, these agents can be characterized as potential *knowledge consumers*. Agents in the group of high-performers possess models which constitute knowledge assets. This situation creates an opportunity to derive a surplus value from the application of these models beyond individual requirements. These agents can decide to act as *knowledge providers* and thus seek to still the demand created by the knowledge consumers.

With a concerted pair of knowledge consumer and knowledge provider which can be understood as complementary knowledge management roles, the minimal role model for agent-oriented distributed knowledge management as proposed in Langer et al. (2006, 2007) has been derived.

Their proposed conceptual framework is now adopted to develop this generic, minimal role model in three consecutive steps (Warden & Herzog 2012). First, concrete versions of the aforementioned core knowledge management roles are derived. In the process, supplemental knowledge management roles are identified which allow for a decomposition of the superordinate knowledge management functions. Second, additional infrastructure roles will be introduced which are instrumental in the match-making process between providers and consumers. In a third step, an additional role is introduced which guides the dynamic adoption of the thus far developed repertoire of main knowledge management roles based on an assessment of the current situation.

### 5.2.1 The Knowledge Consumer Perspective: Advisee and Advice Integration

The development of the necessary role model for multiagent interactive adaptation begins with the consideration of the *knowledge consumer* role.

Langer et al. (2006, p. 7) note that "*an agent acts as a knowledge consumer, the moment it discovers a lack in its own local knowledge repository*". This citation reflects that Langer et al. introduce a notional lack of detailed definition in their use of the term knowledge. Although they use 'knowledge' persistently in their framework, in their case study they consider only the problem of acquiring missing pieces of information in the belief or knowledge base of their agents. These pieces of information qualify as information insofar as they are to be interpreted in the context of deliberation and means-ends reasoning as part of the execution of logistic roles. In the scenario outlined in the preceding section, the notion of a *knowledge consumer* needs to be interpreted as pertaining to knowledge in the form of individual classification models. That is, knowledge which is, for the most part, autonomously induced from observations using machine learning. Hence, knowledge management roles as proposed by Langer et al. (see Figure 5.4 on the next page) are hereafter instantiated on the level of such knowledge.

In contrast to the aforementioned scenario for the adoption of a specific specialisation of the *knowledge consumer* role, the motivation for the scenario presented here is rooted in the self-assessment of a productively used classification model. Instead of a missing piece of knowledge, the problem to be addressed

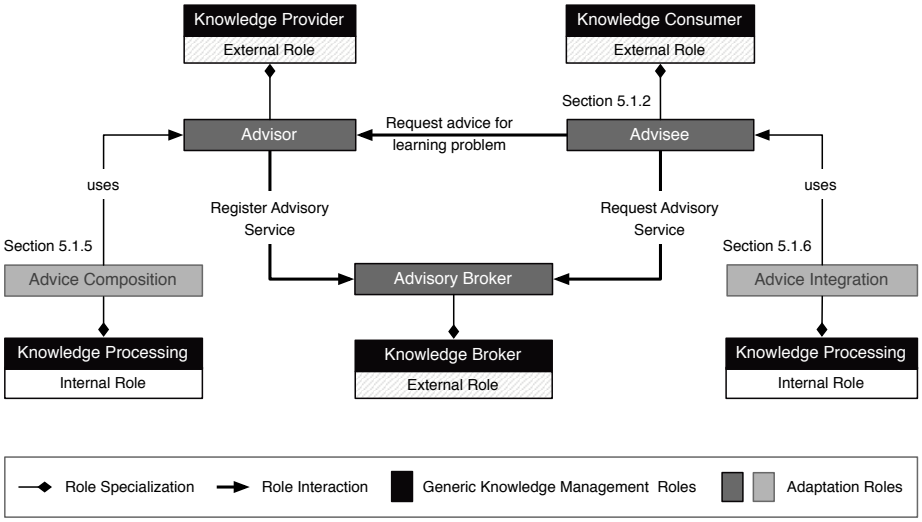



**Figure 5.4:** Interaction-oriented perspective on the composition of dedicated knowledge management roles whose interplay enables an interactive adaptation of individual decision support models in multiagent systems. The adaptation-oriented roles are highlighted. It is also shown how they specialise generic roles that have been proposed for agent-based distributed knowledge management in autonomous logistic processes in (Langer et al. 2006, 2007).

is a non-satisfying performance of a model in its application on novel cases or an independent benchmark data set. The gist in our interpretation of the *knowledge consumer* role is the acquisition of information whose interpretation institutes the opportunity to adapt an original model such that problems in the encoded conceptualisation are concertedly dispelled (the ideal case) or mitigated.

Going forward, the role concept needs to be characterized further to deduce an adequate designation. In line with the proactive disposition of intelligent agents, which is also adopted in machine learning schemes, the actor should actively control its knowledge consumption. This implies that the conduct of this role comprises the identification of helpful information to further the learning process such that well-directed queries can be posed to *knowledge providers*.

The learning setting that is implied with this role complies with a tutorial in academia wherein the student pursues its own learning agenda and actively queries the tutor(s). This setting is consistent with the situation in active learning and argument based machine learning. It differs from scaffolding-based approaches as in active demonstration (cf. Section 4.3.3). Instead of learning by observation and analogy, learning is organised as a discourse structured by iterative queries. The counterpart of this specialisation of the *knowledge consumer* role remains reactive and does not yet on his part shape the model adaptation with an explicit curriculum.

Regardless of the concrete contents exchanged in interaction (which are rendered concrete later in this chapter), it can be argued that the characterisation of the knowledge consumer role justifies to denote it as *advisee role*. Its complement specialisation of the knowledge provider role is consequently denoted as *advisor role*.

### Advisee Role

The *advisee role* is a complex role with regard to necessary reasoning capabilities and deliberation patterns. First, it manages a complete process wherein an agent strives to transition from an original productive model towards a final model with a notably improved performance. As in the iterative knowledge acquisition processes which have been proposed for active learning, a similar multi-tier process needs to be implemented as part of this role. Instead of the simple sequence of assessment-query-response-integration steps, it is proposed to model the process of model adaptation as a search problem. Thus, the exploration of the space of models which are reachable given a) the information acquired from advisors and b) the means to utilise this information is conducted in a well-structured way.

The *advisee role* comprises additional, mandatory functions. First, as Langer et al. (2006, p. 9) state, "*the knowledge consumer has to choose among different knowledge sources, e.g., its own sensors and knowledge by other agents.*" Hence, the *advisee* needs to identify and potentially prioritise available *advisors* to be contacted as knowledge providers over the course of the adaptation process.

Second, the *advisee* needs the ability to assess the momentary model and on that basis identify those knowledge items with a high probability to help achieve the preset adaptation goals. These knowledge items can then be turned into queries which are conveyed to the advisors.
To conclude role responsibilities thus far, the advisee role comprises a) the top-level control for model adaptation, b) the identification of interaction counterparts, and c) the establishment of the knowledge items sought from the advisors.

The elements of the *advisee role* are developed in Section 5.3 (search, identification of interaction partners) and Section 5.4.1 (learning topics). The interaction protocol which structures the interaction between the *advisee* and *advisor role* is introduced in Section 5.4.2.

### Advice Integration Role

A final capability, which is essential to achieve the goals of the advisee role, refers to the utilisation of knowledge items which have been successfully acquired from advisors. Staying within the descriptive framework of Langer et al. (2006) it is feasible to perceive this function as belonging to a subsidiary, but clearly differentiated internal knowledge management role. In fact, it specialises the *knowledge processing role* that "*provides services that generate or reveal new knowledge based on knowledge already available.*"

Different interpretations of this role can be conceived. First, advice provided as integration input may be used to directly revise an existing model, e.g., by pruning or expanding branches in a decision tree, or by a revision of a rule set. A second option to be pursued in the context of this work is to treat the advice integration as a specialisation of the *model acquisition role*. In such a case, a new model is learned based on both the initial training data and thus far procured knowledge items. The latter thereby serve as additional background knowledge to bias/focus the operation of a learning scheme that is able to handle the additional input.

The concrete concept for the *advice integration role* is developed in Section 5.4.6.

### 5.2.2 Knowledge Provider Perspective: Advisor and Advice Composition

#### Advisor Role

The *advisor role* constitutes the role complement of the *advisee role*. It is a specialisation of the generic *knowledge provider* role for which Langer and colleagues note that as part of this role an agent "*provides parts from its internal knowledge repository either on demand or as part of pro-active behavior*" (Langer et al. 2006, p. 8). In the description of the *advisee* role, its proactive character in the pursuit of its associated goals has been highlighted. Specifically, the analogy with an intrinsically motivated student in academia following his/her own agenda to acquire suitable knowledge items has been used. Therefore, the design of the *advisor* role is that of a service-oriented role. Here, the role owner strives to answer advisee queries on-demand. Although the extension to a proactive interpretation of the advisory service is plausible for the future, the concept at hand follows the design decision to concentrate initially on the provision of adequate capabilities with respect to immediate fulfilment of inbound queries.

Such focus is justified as, in contrast to the logistic use case in (Langer et al. 2006), the appropriation of knowledge items that comply with the implicit purpose of advisee queries is a non-trivial operation. It is not a straight-forward retrieval operation on the internal knowledge repository or a mediation process wherein further agents are consulted in order to retrieve knowledge items that are not stored locally. By contrast, the queries posed by the advisee refer to a specific learning problem. Since the advisor possesses a model with the same domain which has a different origin, the task is to address the query based on the agent model. Section 5.4.1 describes that queries ask for rationales or justifications as to *why* an encountered instance should be associated with a preset class within the target concept.

#### Advice Composition Role

In order to reflect that the above interpretation of the *advisor role* involves aspects of knowledge processing, since justifications need to be derived on-demand from an agent's existing knowledge in the form of a suitable decision support model, a dedicated internal *knowledge processing role* is introduced. It is denoted as *advice composition role*. This role serves an important integration function. Specifically, it is designed to act as a layer of abstraction such that the *advisor role* itself can serve requests from advisees without specific knowledge with regard to the particular knowledge representation which is used for the respective advisor model. Since queries are delegated to model-specific implementations of the *advice composition role*, it is already preconceived on the role level, that heterogeneous model landscapes need to be handled in heterogeneous multiagent, and specifically multi-organisational, learning settings as described in Section 5.1.

The concrete concept for the *advice composition role* is developed in Section 5.4.3.

### 5.2.3 Discovery of Advisory Services: Advisory Broker

The knowledge management roles introduced so far were necessary and sufficient in order to support peer-supported model adaptation among an existing group configuration of agents. Specifically, a role apportionment with a single agent in the initiating advisee role and several additional agents in the complementary advisor role has been tacitly assumed.

In dynamic multiagent environments, the discoverability of matching advisory services is an essential prerequisite which enables the formation of temporary knowledge networks in the first place. In general, it is not feasible to configure all potential knowledge consumers with time-invariant knowledge about all eligible knowledge providers. Such a design would obviously forfeit flexibility and autonomy of the knowledge consumers to seek necessary services as the situation arises. In addition, it would also pose severe constraints on the knowledge providers. These ought be allowed to announce or withdraw services from their respective portfolio dynamically, based on individual decisions. Therefore, effective cooperative knowledge management needs suitable infrastructure for service discovery.

For this purpose, Langer et al. (2006, p. 8) propose the introduction of a supplemental *knowledge broker role* in their distributed knowledge management framework. The broker concept is devised in direct analogy to the task-agnostic *yellow page service* which is also part of the FIPA specifications for compliant multiagent platforms (Foundation for Intelligent Physical Agents 2004, p. 6). The authors envision that their knowledge broker not only hosts meta-descriptions of knowledge management services such as knowledge providers, further brokers, or knowledge processing services. They also propose that the broker also maintains a reputation list in order to rank and filter services.

Schuldt (2011) investigates approaches to service discovery in the context of team formation according to the model for cooperation (Wooldridge & Jennings 1999) in multiagent-based autonomous logistics. Here, agents seek to form task-specific teams to jointly achieve a higher degree of fulfilment of logistic objectives in container on-carriage. Since coordination within the teams is essential to pursue joint plans, a high degree of joint commitment is assumed. Teams are represented by a dedicated team manager which serves as a contact for other agents willing to join the team.

Investigated interaction protocols tackle dynamic team formation and maintenance. Their design presupposes different kinds of infrastructures. First, *team formation by directory* assumes a central directory service which administers and, upon request, relays an up-to-date list of team managers. In a subsequent step, all team managers need to be contacted to learn whether their team descriptor matches the individual objectives. This additional indirection in the discovery process is avoided in a second approach, the *team formation by broker*. Therein, both team managers and their team descriptors are administered with a central entity. While the transition from directory to broker decreases interaction com-

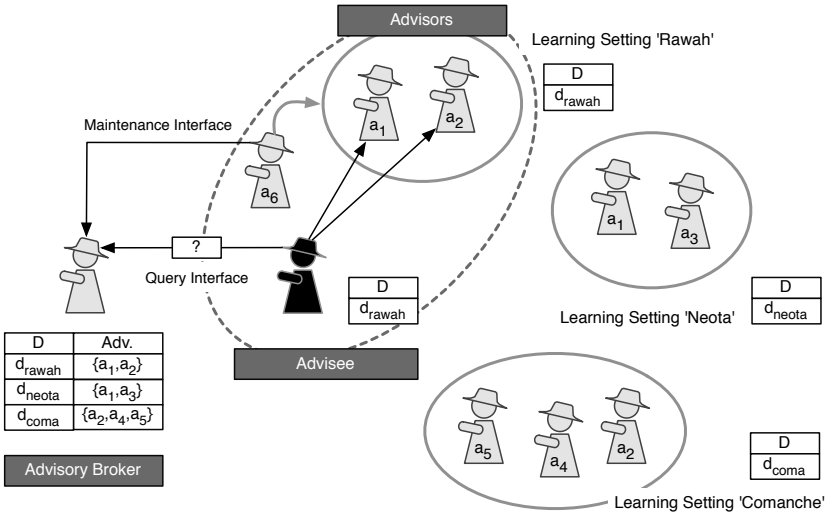



**Figure 5.5:** Concept for a basic advisory broker. It allows agents in the *advisee role* to find agents which offer an advisory service which matches their model descriptor.

plexity, it still retains a central entity. This entity is identified as an interaction bottleneck and a central point of failure (Schuldt 2009). Schuldt also argues that "*applying a broker even aggravates [the directory's] lack of decentralisation because the central broker also makes decisions on behalf of the teams.*" He therefore proposes *team formation by multicast*. While this concept avoids a central entity, it is applicable only if the respective network facilities for multicast services are available.

### Advisory Broker Role

A review of the scenario in Section 5.1 shows that the notion of a team in the interactive adaptation of decision support models does not fit well. The presented advisees do not construe a joint intention and according commitment to achieve a team goal. While this would be the case if they sought to collaborate to form a *global* model as in distributed problem solving, or consolidate distributed local models into a consistent joint model as in argumentation-based multiagent inductive learning (see (Ontañón & Plaza 2010*b*,*d*,*c*)). Instead, agents in the advisee role dynamically enlist advisors in order to pursue their *individual* learning goals. Hence, notions of team and specifically team leader do not apply. As a consequence, the directory approach is not applicable as well.
The concept of a dedicated advisory broker, by contrast, is a feasible option (cf. Fig. 5.5). Rather than administering tuples consisting of team manager and task descriptor, the *advisory broker* in our concept administers a mapping from *model descriptors* to a set of advisors that currently offer a matching advisory service.

The advisory broker is an external role with two interfaces. First, a maintenance interface handles the interaction with advisors that seek to register, de-register, or update their respective service descriptions. Second, a query interface handles service information requests from advisees and allows for placing of subscriptions for notifications on changes in the advisor pool.

It must be guaranteed that the advisory broker role is adopted by a trustworthy

party which is not under suspicion to filter service request in a way that is not intended by an advisee. In intra-organisational learning environments, the *advisory broker role* can be adopted by a regular agent that plans to participate in social knowledge management activities in any form. If the learning setting involves agents from different organisational backdrops, by contrast, an independent, dedicated agent should assume the broker role to dispel privacy concerns.

### 5.2.4 Independent Model Assessment: Benchmark Service

The *advisory broker role* provides the facilities for agents which decided to offer advisory services to have their availability noted by potential clients. It is thereby tacitly assumed that agents have access to sufficient information for informed decisions with regard to the adoption and cancellation of role commitments. Decision-making in this context demands that a knowledge-based agent has access to a service which allows for an independent performance assessment of its productive decision support models that conforms to the following criteria:

a) The assessment refers to the performance of the model on an independent, representative data set. This data set should be endorsed by all participants in interactive model adaptation, both for the advisee (i.e., knowledge consumer) and the advisors (i.e., knowledge provider). As usual in machine learning research, such a data set is referred to as a benchmark data set.

b) The assessment reflects the performance of a model proportional to comparable models from other agents in the same learning setting.

#### Knowledge Benchmarking Role

A *knowledge benchmarking role* which provides a benchmarking service for independent model performance is an essential building block in a comprehensive and self-sufficient role system for our concept for agent-oriented knowledge management.

Unlike the other knowledge management roles introduced so far, the *knowledge benchmarking role* does not specialise a role template suggested by Langer and colleagues. It rather extends their role framework with respect to an independent third-party assessment of an agent's knowledge.

In order provide a benchmark service, the respective agent needs to possess a suitable benchmark data set for each model domain that is addressed within the multiagent system.

Provided that a high level of trust exists in the agent that assumes the *knowledge benchmark role*, these benchmark data sets could be built up dynamically over time with subsets of the data provided as contributions by those agents that seek to use the service. Specifically, each agent designates a sampling from its own data base as its contribution to the benchmark set. This individual contribution reflects a representative set of instances from the point of view of this contributor. In return for the willingness to provide this data, the contributor then qualifies for the use of the benchmark service itself. The more agents agree to these terms,

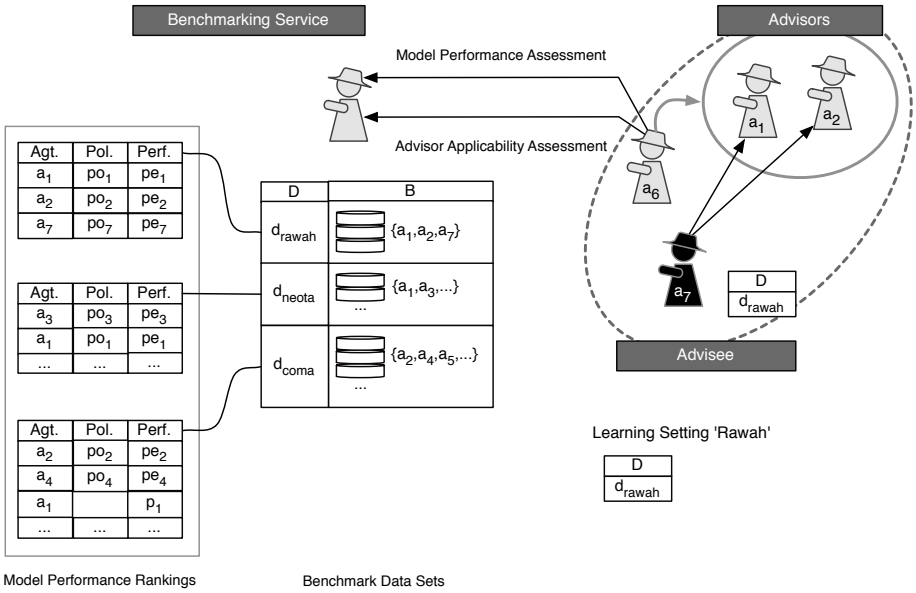



**Figure 5.6:** Concept for a benchmark service. It allows agents to acquire an objective assessment of their respective model performance and support in determining their potential to assume an advisor role.

the more comprehensive and unbiased the benchmark data set becomes and the more objective will the performance assessments become for all participants.

The agent that offers the benchmarking service stores a mapping from model descriptors to benchmark data sets as shown in Figure 5.6. When an agent then requests the assessment of a new model and agrees to expose the inferential abilities of this model for the scope of the interaction, the benchmark service agent can then send unlabelled versions of the instances from the benchmark data set and request their respective classes. The results provided by the client can then be used to compute suitable standard performance indicators.

The scope of the *knowledge benchmarking service* can be taken further. If the responsible agent is allowed to retain results of aforementioned performance assessments, it becomes possible to maintain *domain-specific model rankings*. Thinking back to the conceptual partitioning of agent groups in the same learning setting according to their model performance (cf. Section 5.1), such rankings represent meta-knowledge which allows a potential advisor to determine whether its current model falls into the 'high-performing' group. Consequently, this allows for an informed decision to take on the *advisor role*. If the respective query to the benchmarking service comprises a decision policy, the derived qualitative feedback on advisory applicability can be acquired without exposing specific information about the performance of other agents in the same learning setting. Agents can also place subscriptions with the benchmark service in order to be notified of their advisory applicability status. As a consequence, advisors can keep track coarsely of their model performance and, for instance, understand when its own conceptualisation backslides relative to the group, thereby inducing a dynamic adjustment of the role distribution for a learning setting.

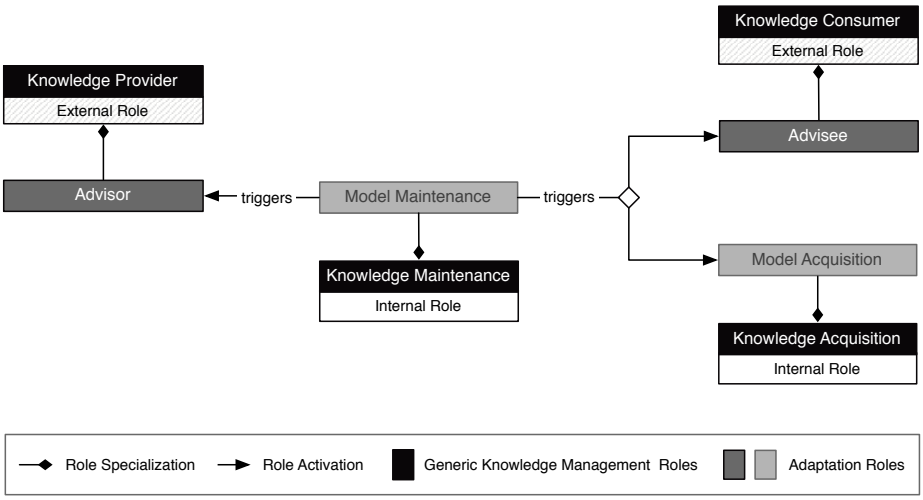



**Figure 5.7:** Control-oriented perspective on dedicated knowledge management roles from the point of view of a single knowledge-based agent. Specifically, the represented roles as a whole can realise the learning element of an agent.

## 5.2.5 Adoption of Knowledge Management Roles: Model Maintenance

The preceding sections have developed a comprehensive role system for agent-oriented knowledge management with a focus on the management of decision support models as complex knowledge items. Initially, the complementary *advisee* and *advisor* roles and their respective subsidiary roles, specifically *advice integration* and *advice composition* have been introduced. These roles constitute the functional core that drives interactive model adaptation. As a next step, the *advisory broker* role to account for advisory discoverability has been introduced. Finally, the *knowledge benchmarking* role was introduced to enable objective model assessment. Even more important however, is the transition to a context specific adoption of the *advisor role*.

### Model Maintenance Role

The final piece in the proposed role system is a knowledge maintenance role. According to Langer et al. (2007), this is an internal role which incorporates tasks to keep the knowledge base manageable and to monitor changes that would trigger further dedicated knowledge management activities. Based on this description, it is feasible to interpret the *model maintenance role* as a top-level role for an agent's knowledge management activities, as shown in Figure 5.7.

Its first function is the continuous assessment of the state of decision support models to be used by the primary domain roles of the agent.

Its second function is the activation of further knowledge management roles dedicated to model acquisition. In this regard, the addition of the *advisee role* increases the scope of action. Such operation flexibility has been characterized

as a critical prerequisite for efficient autonomous control. For instance, with regard to autonomous logistic processes, Windt et al. note that intelligent digital representatives of logistic entities, and thus software agents, need to "*have the methodical capability to use the complete flexibility potential for decision making*" (Windt et al. 2010). While originally an agent was confined to self-sufficient model acquisition, the role system at hand adds a complementary social modality for model adaptation.

A third function of the *knowledge maintenance role* comprises decision-making with respect to a secondary utilisation of local models wherein knowledge-based services are provided for peers in the multiagent system. Within our proposed role system, the *advisor role* constitutes such a service. As stated before, an informed decision about the adoption of such roles presupposes the acquisition of additional meta-knowledge that can be procured by a benchmark service. As a consequence, *model maintenance* is designed as an external knowledge management role with the *benchmark service role* as its complement.

## 5.3 Classification Model Adaptation as Online Search Problem

Within the system of knowledge management roles for the interactive adaptation of individual classification models, the prominent contribution of the advisee role has been emphasised. It is designed to enable and manage the transition from an original productive model to a derived model with a notable increase in predictive performance.

In the following, definitions of concepts that are necessary to characterise the advisee role are procured.

**Definition 5.1** (Data Domain)

*Let $\mathcal{F}_{nom}$ denote the set of nominal features $F_i$ that can be considered in a specific application context where $\forall F_i \in \mathcal{F}_{nom}$ : $\texttt{values}(F_i) := \{f_1, \ldots, f_n \mid n \in \mathbb{N}^+, n > 1\}$ and each $f_j$ stands for some constant symbol. $\mathcal{F}_{num}$, by contrast, denotes the set of respective numerical features. For each such feature $F_i \in \mathcal{F}_{num}$, the possible values range within a certain numeric domain such as $\mathbb{N}$, $\mathbb{R}$ or intervals therein. A feature space $\mathcal{X}^D$ for an application domain is given as an n-tuple of nominal and numeric features.*
*A data domain can then be defined in terms of the selected features space $\mathcal{X}^D$ and a marginal probability distribution for the chosen features $\mathbf{P}(X)$.*

$$Domain := \langle\, \mathcal{X}^D, \mathbf{P}(X) \,\rangle$$

*The set of all data domains will be denoted as $\mathcal{D}$.*

**Definition 5.2** (Classification Task)

*Let a data domain be given as $Domain = \langle\, \mathcal{X}^D, \mathbf{P}(X) \,\rangle$. Let further $Y^D$ denote a special categorical feature for the considered application domain whose values $y \in Y^D$ constitute a partition of the target concept $C$ into unique concept values.*

*A classification task is then defined as a tuple*

$$Task_{Cl} := \langle\, Domain, Y^D, \texttt{classify} \,\rangle$$

*where* $\texttt{classify} : \mathcal{X}^D \rightarrow Y^D$ *is a predictive function. This function is to approximate the underlying true function* $\mathbf{P}(X)$ *such that it becomes possible to predict the class labels of unknown instances* $\mathbf{x} \in \mathcal{X}^D$*.*

**Definition 5.3** (Confusion Matrix)

*Assume a classification problem whose instances are characterised by a vector* $\mathbf{x}$ *in the feature space* $\mathcal{X}^D$*. Let* $\mathcal{H}^D$ *denote the set of possible classification models.* $\mathcal{I}^D$ *denotes a data set of labeled instances, i.e., tuples of the form* $\langle\, \mathbf{x}, y \,\rangle \ : \ \mathbf{x} \in \mathcal{X}^D,\ y \in Y^D$*.*
*The confusion matrix for a model with regard to a data set is then given by a function as follows.*

$$\texttt{confMatrix} : \begin{cases} \mathcal{H}^D \times \mathcal{I}^D & \rightarrow \quad \mathbb{N}_0^{+(n \times n)} \ : \ n = |Y^D| \\ \langle\, M, I \,\rangle & \mapsto \quad CM \end{cases} \tag{5.1}$$

*The resulting matrix is then denoted as* $CM$*. The element* $cm_{i,j}$ *in row* $i$ *and column* $j$ *of the confusion matrix has the value*

$cm_{i,j} := |\{\langle\, \mathbf{x}, y_i \,\rangle \in I \mid \texttt{classify}(M, \mathbf{x}) = y_j\}|$*given an ordering of the* $y \in Y^D$*.*

**Definition 5.4** (Objective Function)

*Let* $M$ *denote a classification model whose target concept has* $n$ *distinct classes. An objective function*[1] *which assesses the model performance is then defined with regard to a fixed data set* $I$*, such as the agent's local test data set as*

$$\texttt{objFunc} : \begin{cases} \mathbb{N}_0^{+(n \times n)} & \rightarrow \quad d \ : \ d \in \mathbb{R},\ 0 \leq d \leq 1 \\ \texttt{confMatrix}(M, I) & \mapsto \quad p_i \end{cases} \tag{5.2}$$

*The set of all valid objective functions is referred to as* $\mathcal{OF}$*.*

---

The basic building block of a comprehensive process of multiagent interactive adaptation conducted by the advisee role is the *model adaptation episode*.
Using the definitions provided above, this concept is defined formally as a function.

**Definition 5.5** (Model Adaptation Episode)

*Let the input classification model that an advisor seeks to improve with respect to an objective function* $\texttt{objFunc}_A$ *be denoted as* $M$*. Let further the learning data set that was used to induce the model be denoted as* $I_{Learn}$*. A pool of background knowledge on the learning task is given as AP for Advice Pool. Finally, a set of advisors as interaction partners for the model adaptation process is given as*

1 For readability, $\texttt{objFunc}(\texttt{confMatrix}(M, I))$ will be abbreviated as $\texttt{objFunc}(M, I)$.

*Advisors. A model adaptation episode is then defined as a function*

$$\texttt{adaptEps} : \begin{cases} \mathcal{H}^D \times \mathcal{I}^D \times \mathcal{AP} \times \mathcal{OF} \times \mathcal{A} & \rightarrow \quad \mathcal{H}^D \times AP \\ \langle\, M, I_{Learn}, AP, \texttt{objFunc}_A, Advisors \,\rangle & \mapsto \quad \langle\, M', AP' \,\rangle \end{cases} \tag{5.3}$$

*where it must hold that*

$$\texttt{objFunc}_A(M, I_{Test}) \leq \texttt{objFunc}_A(M', I_{Test})$$

*The argument tuple for an adaptation episode can be written concisely as AE.*

In Equation (5.3), $AP$ denotes an *Advice Pool*. This concept is defined as a set of pairs as follows.

$$AP := \{\langle\, i, \{LearningAdvice\} \,\rangle\} \;:\; i \in I_{Train} \subset I_{Learn}$$

The concrete structure of learning advice will be deferred until Section 5.4.3. At this point, it suffices to note that a piece of learning advice constitutes background knowledge which relates to a particular instance in the learning data set of an advisee.

The assessment of adaptation success is performed by the advisee based on 1) its custom choice of an objective function $\texttt{objFunc}_A$ and 2) its dedicated test data set $I_{Test} = I_{Learn} \setminus I_{Train}$. It is determined as follows.

$$\texttt{success}(\texttt{adaptEps}(AE))$$

$$= \begin{cases} \textsc{True} & \text{if } \texttt{objFunc}(M, I_{Test}) < \texttt{objFunc}(M', I_{Test})\,, \\ \textsc{False} & \text{else.} \end{cases}$$

The advisee implementation must be designed such that a sequence of subsequent successful model adaptation episodes leads to a monotonous increase in the performance of the resulting models until no further improvement can be reached.

Definition 5.5 has provided but a signature of the complex control process for interactive model adaptation. To fill the definition with life, modeling of the adaptation as a search problem is proposed. Such an approach is well-known in the machine learning community. Separate-and-conquer algorithms, for instance, use beam search algorithms in order to search the space of rule complexes. In contrast to such examples, multiagent interactive adaptation cannot be treated as a traditional example of offline search. Although this is not directly conveyed by Definition 5.5 alone, the high-level description of the learning problem in the initial section of this chapter has shown that both the identification of advisors and subsequent interaction with them for the purposes of advice acquisition constitute integral parts of the adaptation process. Hence, ex-ante, an agent does not yet possess the complete information that is required to compute a complete solution to the search problem.

The agent rather needs to interleave phases of computation with action execution. As a general pattern, a decision about the immediate next action is computed. The action is then executed, and its effects in the environment are observed. Only based on these observations can than another action be computed. The search

process thus is an online search problem.

Online search is described as a suitable strategy in order to cope with dynamic or semi-dynamic domains where the decision for the next action needs to be made fast. In non-deterministic environments, the stepwise progress of the search allows for handling contingencies as they materialise. What is more important with respect to the application of online search at hand is that it constitutes a necessary measure in unknown environments. Here, an agent cannot know right from the start the scope of the search space and specifically what effects its actions will have. It then faces an exploration problem. The actions that it engages in during search have the character of experiments. They contribute to the acquisition of necessary additional input, such that informed choice of the next step is enabled.

### 5.3.1 Characterisation of the Search Problem

In the following, a comprehensive characterisation of the online search problem for model adaptation is developed, beginning with the definition of the initial search state.

**Definition 5.6** (Initial Search State)

*Let the initial classification model whose performance according to an objective function is to be increased be denoted as $M_{Orig} \in \mathcal{H}^D$. Let further the set of learning instances be denoted as $I_{Learn} \in I^D$. The initial advice pool is given as $AP$. Finally, let $ES$ denote an additional data structure, defined below in Definition 5.8, which is used to record the progress of the expansion of the search state.*
*The initial state for the search problem is then defined as a tuple*

$$S_{Init} := \langle M_{Orig}, I_{Learn}, AP, ES \rangle. \tag{5.4}$$

*The set of all reachable states for the online search problem is denoted as $\mathcal{S}$ and will itself be introduced in Definition 5.10 on page 109.*

**Definition 5.7** (Model Adaptation Step)

*The transition between search states in the context of multiagent interactive adaptation is denoted as adaptation step.*

**Definition 5.8** (Expansion State for Search States)

*Let $LP_{prop} \subset I_{Train} \in \mathcal{I}^D$ constitute a set of learning problems for whom it has been proposed that they should be addressed in the context of an adaptation step. Let further $LP_{reject}$ be a set of learning problems for which is has been ascertained that they cannot be addressed with the current advisor set. Finally, $LP_{accept}$ denotes a set of pairs whose first element $lp \in LP_{prop} \setminus LP_{reject}$ and whose second element is learning advice acquired to address the respective learning problem.*

*The expansion state for a search state is then defined as a tuple*

$$ES := \langle\, LP_{prop}, LP_{reject}, LP_{accept} \,\rangle. \tag{5.5}$$

Starting from the initial state, it is necessary to *expand* the current state by applying each legal action that is defined for the search problem.

When entering a new search state, it holds that $ES = \langle\, \emptyset, \emptyset, \emptyset \,\rangle$. The expansion of the current search state for the purpose of model adaptation is in itself a three-tiered process whose progress is tracked with the expansion state. Only when $LP_{accept}$ has been acquired a set of successor states can be generated. State expansion is also an interactive process of knowledge acquisition, as it incorporates the communication among the advisee and advisor roles outlined in Section 5.2.

**State Expansion: Propose Learning Topics** The first step in state expansion involves an assessment of the model. The goal of this operation is the identification of weaknesses in the model's conceptualisation. These show themselves, for instance, in frequent incorrect classifications of training instances in repeated internal cross-validations. Based on identified model deficiencies, a set of *learning topics* $lp \in LP_{prop} \subset I_{Train}$ is suggested.

$$\texttt{propExpand} : \begin{cases} \mathcal{H}^D \times \mathcal{I}^D & \rightarrow \quad d \;:\; d \subset \mathcal{I}^D \\ \langle\, M, I_{Train} \,\rangle & \mapsto \quad LP_{prop} \;:\; |LP_{prop}| \ll |I_{Train}| \end{cases} \tag{5.6}$$

The exact procedure is described in Section 5.4.1. Proposed learning topics can be conceived as potential points of origin for the expansion of the current state. Their total amount $bf_p = |LP_{prop}|$ indicates the upper bound for the effective branching factor.

**State Expansion: Ascertainment of Options** In order to dispel uncertainty whether a learning topic qualifies as a valid expansion point, the advisee needs to engage in knowledge acquisition. In fact, for each potential learning problem $lp \in LP_{prop}$, the advisee interacts with a set of advisors ($Advisors$). Therein, it seeks learning advice for the presented topic. The concrete interaction protocol that structures the role interaction between advisee and advisor is presented in Section 5.4.2.

$$\texttt{checkExpand} : \begin{cases} LP_{prop} & \rightarrow \quad \mathbb{B} \times LearningAdvice \;:\; LP_{prop} \subset \mathcal{I}^D \\ lp & \mapsto \quad \langle\, \texttt{canAdvise}(Advisors, lp), \\ & \qquad\quad \texttt{getAdvice}(Advisors, lp) \,\rangle \end{cases} \tag{5.7}$$

Each attempt to acquire learning advice for a learning topic bears the contingency of failure due to a number of reasons. On the syntactic level, the query that is prepared by the advisee may be incomprehensible for the advisor. On the organisational level, a comprehensible query may be rejected due to insufficient authorisation or motivation to cooperate. Finally, on the semantic level, even when an advisor has understood a query and has made a commitment to cooperate, its

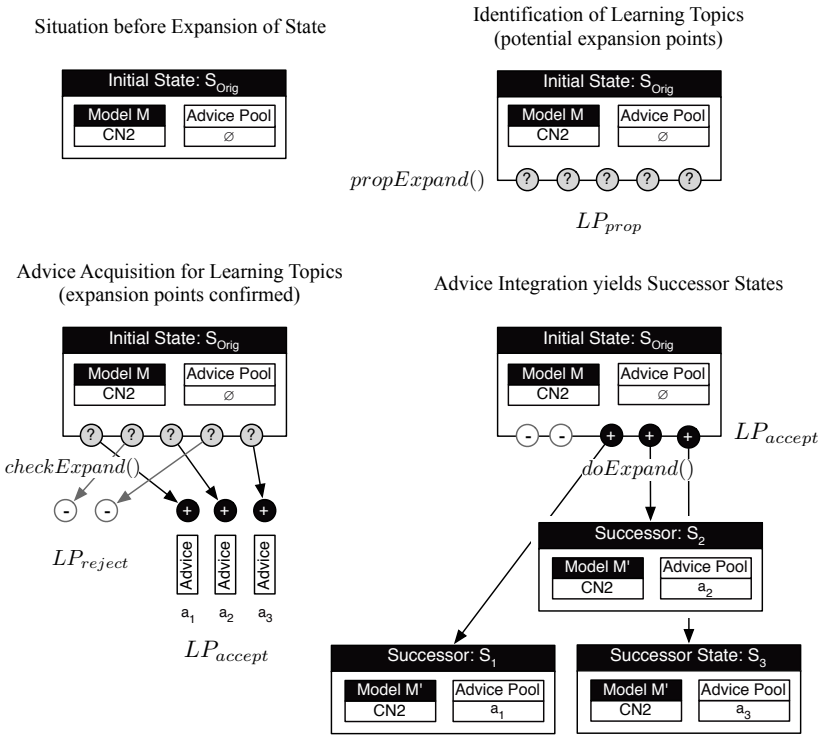



**Figure 5.8:** *From top-left to bottom right*: Schematic overview of the consecutive stages of a state expansion in the presented online search for improved classification models. Steps three is especially important, as it involves engaging in interaction with suitable advisors in order to acquire learning advice which constitutes the basis for the computation of successor models and thus, states.

own model may not afford the computation of the desired learning advice. This can happen, when the advisee model is afflicted with an issue in prediction that is similar to the one addressed in the current interaction protocol invocation.

When the advice acquisition fails for one of the reasons enumerated above such that $\texttt{canAdvise}(Advisors, lp) = \textsc{False}$, the expansion state is updated as

$$ES' \leftarrow \langle\, LP_{prop} \setminus lp^{-}, LP_{reject} \cup lp^{-}, LP_{accept} \,\rangle.$$

If, however the acquisition process can be completed successfully, such that $\texttt{canAdvise}(Advisors, lp) = \textsc{True}$, the expansion state is updated as

$$ES' \leftarrow \langle\, LP_{prop} \setminus lp^{+}, LP_{reject}, \texttt{join}(LP_{accept}, \langle\, lp^{+}, LearningAdvice \,\rangle) \,\rangle$$

where $LearningAdvice = \texttt{getAdvice}(Advisors, lp^{+})$.

The advisee conducts a sequence of interactions, i.e., exploration actions, until $LP_{prop} = \emptyset$. To conclude, the sequence of interactions with the advisors serve a dual purpose. With each unsuccessful attempt at knowledge acquisition, the according learning topic can be excluded from expansion of the current search state.

In the end, the set $LP_{accept}$ enumerates the subset of learning topics for which additional advice (with respect to that already stored in the advice pool $AP$)

could be obtained, thus enabling an actual extension of the current state. Hence, $bf_e = |LP_{accept}|$ constitutes the actual branching factor at the current state.

**Execution of State Expansion** For the actual expansion of the current state, the advisee employs the capabilities of its internal advice integration role. From the point of view of the online search, this role effectively provides a function which takes the current state and the acquired advice for a learning problem as input, computes a new model, and subsequently updates the advice pool.

$$\texttt{doExpand} : \begin{cases} \mathcal{H}^D \times \mathcal{I}^D \times \mathcal{AP} \times \mathit{LearningAdvice} & \rightarrow & \mathcal{H}^D \times \mathcal{AP} \\ \langle\, M, I_{Learn}, AP, \mathit{LearningAdvice} \,\rangle & \mapsto & \langle\, M', AP' \,\rangle \end{cases} \tag{5.8}$$

Based on this result, the successor state is then given as

$$S_{Succ} := \langle\, M', I_{Learn}, AP', \langle\, \emptyset, \emptyset, \emptyset \,\rangle \,\rangle$$

while the complete set of successor states is denoted as $\mathit{Successors}$.

Once a successor model has been computed for each learning topic, the frontier of child nodes that can be reached from the current state is complete.

In contrast to conventional search problems, the actions that are applicable in a search state are actually composite actions. Knowledge acquisition is employed first as a filter for valid actions contingent on the respective advisors. Subsequently, the advice integration role accounts for the actual executable actions.

In terms of modelling the search problem at hand, the composite actions signify possible actions available to the agent, while the design of the advice integration role, presented in Section 5.4.6, specifies the state transition model. Taken together, the initial state, actions, and the transition model define the reachable region of the global state space $\mathcal{S}$ for the search problem.

Generally speaking, this would be the subset of all states which can be reached from the initial state by the application of a certain action sequence. Relating to the concrete search problem, the state space is hence the set of all classification models that are reachable from the original model by a sequence of adaptation steps where each step is defined by a) the addressed learning problem, b) the interaction partner(s) engaged in knowledge acquisition, c) their respective model and method to procure model-based learning advice, and d) the method for advice integration.

The scope and breadth of the exploration of the theoretically reachable region of the global search space $\mathcal{S}$ itself, and along with it the optimality the exploration, is then contingent on the search algorithm.

**Definition 5.9** (Direct State Reachability)

*Let $S_i = \langle\, M_i, I_{Learn}, AP_i, \langle\, \emptyset, LP_{reject}, LP_{accept} \,\rangle \,\rangle$ denote an expanded search state. Then, another search state $S_j = \langle\, M_j, I_{Learn}, AP_j, ES_j \,\rangle$ in the state space $\mathcal{S}$ is directly reachable from $S_i$ iff*

$$\exists\, \langle\, lp_k^+, \mathit{LearningAdvice}_k \,\rangle \in LP_{accept} :$$

$$\langle\, M_j, AP_j \,\rangle = \texttt{doExpand}(M_i, I_{Learn}, AP_i, LearningAdvice_k)$$

*$S_j$ is then a direct successor state created through expansion of $S_i$ such that $j = i + 1$. When state $S_{i+1}$ is directly reachable from state $S_i$, the notation $S_i \succ S_{i+1}$ is used.*

Based on the initial state of the presented search problem in Definition 5.6 on page 105 and the notion of direct state reachability from Definition 5.9, it is now possible to finally specify the search space for multiagent interactive adaptation constructively.

**Definition 5.10** (Model Adaptation State Space)

*Let the initial state for a process of multiagent interactive model adaptation, conforming to Definition 5.6, be denoted as $S_{Orig}$. Then the state space for the search is defined constructively as follows.*

$$\mathcal{S} := \{S_i \mid S_i = S_{Orig} \vee \exists\, S_1, \ldots, S_n \;:\; S_1 = S_{Orig},\, S_n = S_i,\, S_j \succ S_{j+1}\}$$

The above definition of the state space for multiagent interactive adaptation highlights the significance of the *pre-set* initial state $S_{Orig}$. For the online search problem at hand, this has consequences with regard to the applicable search algorithms.

In particular, algorithms that employ random restarts, such as *random restart hill climbing*, are not applicable. Their assumption is that in the given application scenario, it is feasible from both a technical and cost perspective, to compute multiple seed states which are ideally located in different regions of the state space. Then a sequence of individual searches can be conducted where each search instance uses another seed state as its initial state. The rationale for this approach is that the likelihood is increased with each additional search that this instance converges to another local optimum (which may also be the global optimum). Hence, post-hoc selection of the end result of the best search instance leads to an overall improved result compared with a single search run.

In the search problem presented here, the original model of the advisor is pre-set. Learned from training data and an original advice pool, it constitutes the highest-performing model the agent could obtain thus far, either as the result of self-sufficient learning activities or previous cycles of interactive multiagent adaptation.

Consequently, a non-restarting hill climbing search as shown in Algorithm 1 is used, which can be extended to a stochastic hill climbing algorithm such as a version of simulated annealing, and also adopt local beam search.

**Successor State Selection** The confusion matrix, as measured against the – with respect to the training data – independent and for the the duration of an adaptation episode also time-invariant test data set of an advisee agent constitutes the basis of an objective function that is agnostic of the respective learning task. For binary classification tasks, one basic example for such an objective function would be the classification *accuracy*, i.e., the ratio of correctly classified instances in the test data set (the diagonal of the confusion matrix) on all instances.

**Algorithm 1** Online search for a model adaptation episode, implemented in the advisee role played by agent $A$.

**Input:** $M \in \mathcal{H}^D$ : Initial classification model, subject to adaptation in the episode
$I_{Learn} \in \mathcal{I}^D$ : Learning data for the classification task
$AP \in \mathcal{AP}$ : Initial pool of learning advice
$\texttt{objFunc}_A \in \mathcal{OF}$ : Custom objective function to measure model performance
$Advisors \subset \mathcal{A}$ : Advisors that are to contribute advice for learning problems
**Output:** $M' \in \mathcal{H}^D$ : Final model returned by the online search
$AP' \in \mathcal{AP}$ : Extended pool of learning advice after adaptation episode

1: **function** ADAPTEPISODE($M, I_{Learn}, AP, \texttt{objFunc}_A, Advisors$)
2: $S_{Init} \leftarrow \langle\, M, I_{Learn}, AP, \langle\, \emptyset, \emptyset, \emptyset \,\rangle \,\rangle$
3: $S_{Curr} \leftarrow S_{Init},\ tryClimb \leftarrow$ TRUE
4: **while** ($tryClimb =$ TRUE) **do**
5: $Successors \leftarrow \emptyset,\ tryClimb \leftarrow$ FALSE
6: $LP_{open} \leftarrow \texttt{propExpand}(S_{Curr}(M), I_{Train})$ % see Equation (5.6)
7: $S_{Curr}(ES) \leftarrow \langle\, LP_{open}, \emptyset, \emptyset \,\rangle$
8: **for** ($lp\ :\ LP_{open}$) **do**
9: $\langle\, Success, LearningAdvice \,\rangle \leftarrow \texttt{checkExpand}(lp)$ % see Equation (5.7)
10: **if** ($Success =$ TRUE) **then**
11: $S_{Curr}(ES) \leftarrow \langle\, LP_{prop} \backslash lp, LP_{reject}, \texttt{join}(LP_{accept}, \langle\, lp, LearningAdvice \,\rangle) \,\rangle$
12: **else**
13: $S_{Curr}(ES) \leftarrow \langle\, LP_{prop} \setminus lp, lp_{reject} \cup lp, LP_{accept} \,\rangle$
14: **for** ($(lp, advice)\ :\ LP_{accept}$) **do**
15: $\langle\, M', AP' \,\rangle \leftarrow \texttt{doExpand}(S_{Curr}(M), I_{Learn}, S_{Curr}(AP), advice)$ % Eq. (5.8)
16: $Successors \leftarrow Successors \cup \{\langle\, M', I_{Learn}, AP', \langle\, \emptyset, \emptyset, \emptyset \,\rangle \,\rangle\}$
17: $S_{Next} \leftarrow \texttt{bestSuccessor}(Successors, \texttt{objFunc}_A)$
18: **if** ($\texttt{objFunc}_A(S_{Curr}(M)) \leq \texttt{objFunc}_A(M')$) **then**
19: $S_{Curr} \leftarrow S_{Next}$
20: $tryClimb \leftarrow$ TRUE
21: **return** $\langle\, S_{Curr}(M), S_{Curr}(AP) \,\rangle$

A large body of alternative measures has been introduced. For instance, Sokolova & Lapalme (2009) present an a systematic analysis of key performance indicators for classification tasks. One dimension in their systematisation of objective functions is the type of classification problem at hand. They distinguish binary classification, for which the introductory *accuracy* can be employed besides more sophisticated alternatives, as well as multi-class classification, multi-label classification and finally hierarchical classification.

### Instantiations of the Objective Function

In the following, the focus will be put on the multi-class case that is quite common in real world classification tasks. Hence, several paradigmatic objective

functions which are adequate for the presented model optimisation problem are enumerated. In principle, it should be noted that individual agents may have different preferences with respect to the improvement of their original model. Hence it is necessary to characterise the objective functions with respect to their bias, such that an adequate choice for a given model adaptation scenario can be found.

**Example 1: Objective Function based on Average Accuracy** The *average accuracy* of a model, measured against a test data set, provides a straight forward means to define an objective function according to Equation 5.2 as follows.

$$\texttt{objFunc}_{Avg\_Acc} : \begin{cases} \mathbb{N}_0^{+(n\times n)} & \rightarrow \quad d \ : \ d \in \mathbb{R},\ 0 \leq d \leq 1 \\ \texttt{confMatrix}(M, I_{Test}) & \mapsto \quad p_{Avg\_Acc} \end{cases} \tag{5.9}$$

The confusion matrix for a classification model $M$ and a test data set $I_{Test}$ is denoted as $CM$ such that $CM = \texttt{confMatrix}(M, I_{Test})$. The elements of $CM$ are given in lowercase with line and column index, as in $cm_{i,j}$. The value $p_{Avg\_Acc}$ of the objective function based on accuracy is then given as

$$p_{Avg\_Acc} : CM \mapsto \frac{\sum_{i=1}^{l} \frac{tp_i + tn_i}{tp_i + tn_i + fn_i + fp_i}}{l}$$

The evaluation focus of the *average accuracy* is the average per-class effectiveness of a classifier. It falls into the macro-averaging category of performance measures and gives equal weight to each value of the target class. Hence, in case of an uneven class distribution in the data set fuelling the confusion matrix, the minority classes are treated as having equal importance to the majority class.

In relation to the model optimisation goal, this means that by using *average accuracy* as an objective function, the advisee agent seeks to improve classification equally across all classes.

**Example 2: Objective Function based on Matthews Correlation Coefficient** A second example for an objective function that has been extended successfully for use with multi-class classification is the Matthews Correlation Coefficient (MCC) (Gorodkin 2004). In its binary form, this measure is also known in the literature as $\phi$-coefficient. For binary classification tasks, Jurman and colleagues note that the MCC has been well-received by the machine learning community due to its capability to summarise the confusion matrix into a single scalar value (Jurman et al. 2012). The authors also add that the MCC is used as a reference performance measure also for unbalanced data sets in practical fields such as bioinformatics. The generalised form of the MCC was introduced by Gorodkin (2004) in exactly that field to evaluate predicted k-category assignments of biological sequences.

Given a classifier based on a model $M$ and a number of samples $S$ in a test data set, let $X, Y \in M(I \times N, \mathbb{F}_2)$ be two matrices. The first matrix $X$ represents a ground truth for the samples $s \in I_{Test}$ such that $x_{sn} = 1 \iff s = \langle\, \mathbf{x}, n \,\rangle$, and otherwise

$x_{sn} = 0$. The second matrix Y, by contrast represents the class assignments for the classifier under test. Hence, $y_{sn} = 1 \iff \texttt{classify}_M(\mathbf{x}) = n$, and otherwise $x_{sn} = 0$.

Let further $CM \in M(N \times N, \mathbb{N}) = \texttt{confMatrix}(M, I_{Test})$ denote the corresponding confusion matrix. The Matthews Correlation Coefficient is then defined by the following equation in terms of covariances.

$$MCC = \frac{COV(X,Y)}{\sqrt{COV(X,X) \cdot COV(Y,Y)}} \tag{5.10}$$

The covariance between the matrices $X, Y$ is then determined as follows

$$COV(X,Y) = \sum_{k=1}^{N} w_k \cdot COV(\mathbf{x}_k, \mathbf{y}_k) \tag{5.11}$$

where $w_k = \frac{1}{N}$. Also, the covariance $COV(\mathbf{x}_k, \mathbf{y}_k)$ can be recast as follows.

$$COV(X,Y) = \sum_{k=1}^{N} \frac{1}{N} \cdot (\sum_{s=1}^{S}(x_{sk}) - \bar{x_k}) \cdot (y_{sk} - \bar{y_k}) \tag{5.12}$$

It holds that $\bar{x_k} = \frac{1}{S}\sum_{s=1}^{S} x_{sk} = \frac{1}{S}\sum_{l=1}^{N} c_{kl}$, and $\bar{y_k} = \frac{1}{S}\sum_{s=1}^{S} y_{sk} = \frac{1}{S}\sum_{l=1}^{N} c_{lk}$. Using these facts and entering Equation (5.12) into Equation (5.10), the Matthews Correlation Coefficient can be rewritten as:

$$MCC = \frac{\sum_{k,l,m=1}^{N} c_{kk}c_{ml} - c_{lk}c_{km}}{\sqrt{\sum_{k=1}^{N}[(\sum_{l=1}^{N} c_{lk})(\sum_{f,g=1:f\neq k}^{N} c_{gf})]}\sqrt{\sum_{k=1}^{N}[(\sum_{l=1}^{N} c_{kl})(\sum_{f,g=1:f\neq k}^{N} c_{gf})]}}$$

With this form, the calculation of the Matthews Correlation Coefficient has been rebased directly on the regular confusion matrix $CM \in M(N \times N, \mathbb{N})$ rather than the original matrices $X, Y \in M(I \times N, \mathbb{F}_2)$.

MCC has the co-domain $[-1, 1]$ where 1 signifies a perfect classification. The opposite extreme of -1 is reached asymptotically in the extreme misclassification case of a confusion matrix with all zeros but in two symmetric entries $c_{ij}$ and $c_{ji}$.

Jurman and colleagues note that "*MCC is a good compromise among discriminacy, consistency and coherent behaviors with varying number of classes, unbalanced datasets, and randomization. [. . .] Furthermore, the behaviour of the MCC remains consistent between binary and multiclass settings.*"(Jurman et al. 2012, p. 7). They conclude, that for confusion matrix analyses, MCC can be considered a suitable off-the-shelf solution across many practical tasks.

As illustrated in Definition 5.5 on page 104, a suitable objective function which accords with individual learning preferences needs to be chosen by the advisee agent to determine the winning successor state from the set *Successors* of potential candidates. For standard, i.e., non-stochastic hill climbing search, the selection of the successor model is calculated as follows (see Algorithm 1, line 17).

**Algorithm 2** Control loop for multiagent interactive adaptation, implemented in the advisee role played by agent $A$.

**Input:** $Task \in \mathcal{T}$ : Descriptor for the addressed classification problem
$M_{Orig} \in \mathcal{H}^D$ : Initial classification model, subject to adaptation
$I_{Learn} \in \mathcal{I}^D$ : Learning data for the classification task $Task$
$AP_{Orig} \in \mathcal{AP}$ : Initial pool of learning advice
$\texttt{objFunc}_A \in \mathcal{OF}$ : Custom objective function to measure model performance
**Output:** $M_{Adapt} \in \mathcal{H}^D$ : Final model returned by the adaptation process
$AP_{Adapt} \in \mathcal{AP}$ : Extended pool of learning advice after adaptation

1: **function** INTERACTIVEADAPTATION($M_{Orig}, I_{Learn}, AP_{Orig}, \texttt{objFunc}_A, Task$)
2: $\langle M_{Adapt}, AP_{Adapt} \rangle \leftarrow \langle M_{Orig}, AP_{Orig} \rangle$
3: $Advisors \leftarrow \texttt{acqAdvisors}(Task, \texttt{objFunc}_A)$
4: **while** ($Advisors \neq \emptyset$) **do**
5: $Advisors_{Ep} \leftarrow \texttt{selectForEpisode}(Advisors)$
6: **repeat** % Model Adaptation Episode
7: $\langle M', AP' \rangle \leftarrow$ ADAPTEPISODE($M_{Succ}, I_{Learn}, AP_{Adapt}, \texttt{objFunc}_A, Advisors_{Ep}$)
8: **until** ($\texttt{objFunc}_A(M', I_{Test}) \leq \texttt{objFunc}_A(M_{Adapt}, I_{Test})$)
9: **if** ($\texttt{objFunc}_A(M', I_{Test}) \leq \texttt{objFunc}_A(M_{Adapt}, I_{Test})$) **then**
10: $Advisors \leftarrow Advisors \setminus Advisors_{Ep}$
11: **else**
12: $\langle M_{Adapt}, AP_{Adapt} \rangle \leftarrow \langle M', AP' \rangle$
13: **return** $\langle M_{Adapt}, AP_{Adapt} \rangle$

$$\texttt{bestSuccessor} : \begin{cases} \mathcal{P}(\mathcal{S}) \times \mathcal{OF} & \rightarrow \quad \mathcal{S} \\ \langle Successors, \texttt{objFunc}_A \rangle & \mapsto \quad S_{Next} \end{cases} \tag{5.13}$$

where $S_{Next} = \underset{S_i \in Successors}{Arg\,max} \; (\texttt{objFunc}_A(S_i(M), S_i(I_{Test})))$.

### 5.3.2 Choice of Advisors

Definition 5.5 on page 104 highlighted that a proper set of advisor agents is a mandatory pre-requisite for the conduct of a model adaptation episode. In particular, in the context of state expansion, Equation (5.7) has shown that the knowledge acquisition from a momentary subset of advisors is the essential enabler for state expansion.

In the spirit of an autonomous shaping of the individual adaptation path, the set of available advisors is not treated as a fixed input to the adaptation process. The advisor panel is rather assorted and actively maintained by the advisee itself as part of its role.

In the initialisation phase of the adaptation process, i.e., before the begin of

any search activity in the context of an adaptation episode, the advisee engages in an interaction with the dedicated agent that fills the advisory broker role (cf. Section 5.2.3 on page 97) to learn about available advisors that provide their services for the targeted classification task. The acquired advisor set is then broken into advisor panels. Formally, this corresponds to the following function.

$$\texttt{acqAdvisors} : \begin{cases} \mathcal{T} \times \mathcal{OF} & \rightarrow \quad \mathcal{P}(\mathcal{A}') \ : \ \forall A \in \mathcal{A}' : \texttt{playsRole}(A, \textsc{Advisor}) \\ \langle\, Task, \texttt{objFunc}_A \,\rangle & \mapsto \quad Advisors = \{Adv_i \mid Adv_i \in \mathcal{P}(\mathcal{A})\} \end{cases}$$

where $\forall A \in Adv_i \ : \ \texttt{doesAdvise}(A, domain)$.

Once the set $Advisors$ has been acquired, and in case $Advisors \neq \{\emptyset\}$, the advisee can engage in an adaptation episode according to Definition 5.5 using its current model, advice pool, and advisor panel given as $Adv_i \in Advisors$. If the adaptation episode is successful, such that the executed online search process has yielded an improvement in model performance as measured by the objective function $\texttt{objFunc}_A$ against the test data set $I_{Test}$ and $|Advisors| > 1$, the successful advisor panel is retained. Otherwise, it is removed from the set.

While $|Advisors| \geq 2$, the adaptation process continues with a subsequent adaptation episode. If the advisor panel for this episode is denoted as $Adv_i \in Advisors$, then it must hold that $Adv_{i-1} \neq Adv_i$. This has also been illustrated in Algorithm 2.

The rationale here is, that in the preceding adaptation episode, the agents in $Adv_{i-1}$ have already contributed to their best knowledge to the improvement of the advisor model. Since then, the situation has not yet changed such that an immediate repeated consultation is not promising. Therefore, it is guaranteed, that other advisors are engaged first to give them a chance to contribute to another change of the current model.

### 5.3.3 Definition of Interactive Multiagent Adaptation

To conclude, the full process of multiagent interactive adaptation is in general designed as a sequence of consecutive adaptation episodes.

**Definition 5.11** (Multiagent Interactive Adaptation)

*Let the initial classification model that an advisee A seeks to improve with regard to its objective function* $\texttt{objFunc}_A$ *be denoted as* $M_{Orig}$*. The grounds on which this model has been learned, is given by the learning data* $I_{Learn}$ *and an initial – potentially empty – advice pool* $AP_{Orig}$*.* $Adv_i$ *denotes the advisory panel for the respective adaptation episode* $AE_i$*. A comprehensive process of multiagent interactive adaptation is then defined as a sequence of consecutive adaptation episodes as follows.*

$$MIA := (AE_1, \ldots, AE_n) \ : \ n \in \mathbb{N}^+ \tag{5.14}$$

*where it must hold that*

$$(1) \quad AE_1 = (M_{Orig}, I_{Learn}, AP_{Orig}, \texttt{objFunc}_A, Adv_1),$$
$$(2) \quad (AE_{i+1}.M, AE_{i+1}.AP) = \texttt{adaptEpisode}(AE_i),$$
$$(3) \quad AE_n.M' = M_{Adapt}.$$

*The final model $M_{Adapt}$ is then the replacement for the initial model $M_{Orig}$ and can be operationalised as new productive model.*

## 5.4 Building Blocks for Classification Model Adaptation

The preceding section has formally introduced the general framework for multiagent interactive adaptation from the point of view of the central advisee knowledge management role. This section now fleshes out important aspects of the adaptation process.

### 5.4.1 Identification of Learning Problems

In Section 5.3.1, it was shown that the first step in the process of state expansion in the context of the online search procedure of an adaptation episode is the proposal of suitable learning problems. To that end, the function `propExpand` : $\mathcal{H}^D \times \mathcal{I}^D \rightarrow d : d \subset \mathcal{I}^D$ has been introduced (see Equation (5.6) on page 106).

The basic question which guides the procurement of suitable learning problems is: *For which instances within the training set used to induce a classification model is the acquisition of instance-related background knowledge the most promising with regard to resulting model improvements?* One answer to this question, which has been brought up in the literature (Možina et al. 2007), associates the critical nature of training instances with the number of incorrect classification attempts within a repeated internal cross-validation procedure.

Specifically the training instances are used as input data for a $k$-fold cross-validation, where the input sample is randomly partitioned into $k$ subsamples or folds. Subsequently $k$ different settings are constructed for the learner. In each setting, the instances of a different single fold are retained as test or validation set, while the remaining $k-1$ folds constitute the training set. When the cross-validation is conducted, each instance from the original training set of the advisee appears exactly once in an internal test set. Thus for each $k$-fold cross-validation, each sample is either classified correctly or incorrectly exactly once. If the $k$-fold cross-validation is repeated $n$ times with a different random seed, this will result in different folds to be generated. If the number of incorrect classifications for all training instances are thus accumulated over $n$ subsequent cross-validation runs, effects that arise from a specific fold structure are extenuated. An ordering of examples by the number of incorrect classifications in combination with a significance threshold $t_{min}$ on the error count thus leads to a basic sets of so-called critical instances. This can be formalised as follows.

**Definition 5.12** (Set of Critical Instances)

*Let* $\mathbf{1}_{\mathcal{X}_s} : \mathcal{I}^D \rightarrow \{0,1\}$ *denote an indicator function for the correctness of classification decisions on samples* $\langle x, y \rangle \in I_{Train}$ *during a k-fold cross-validation* $\mathcal{X}_s(I_{Train})$ *with a seed* $s \in \mathcal{S}$ *such that*

$$\mathbf{1}_{\mathcal{X}_s}(x) := \begin{cases} 1 & if \quad \texttt{classify}(x) \neq y : \langle x, y \rangle \in I_{Train} \\ 0 & if \quad \texttt{classify}(x) = y. \end{cases} \tag{5.15}$$

*Let further* $S : |S| = n$ *denote a set of distinct random seeds for repeated cross-validation runs and let* $t_{min} \in \mathbb{N}^+$ *be a significance threshold.*

*The set* $LP_{prop} \subset I_{Train}$ *of critical instances is then defined as*

$$LP_{prop} := \{\langle x, y \rangle \in I_{Train} : \sum_{i=1}^{n} (\mathbf{1}_{\mathcal{X}_{s_i}}(x)) \geq t_{min}, s_i \in S\}. \tag{5.16}$$

Given that $\mathcal{H}^D$, via incorporated background-knowledge, is reflected in $\mathcal{X}_s(I_{Train})$, the function `propExpand` is thus implemented as

$$\texttt{propExpand} : \begin{cases} \mathcal{H}^D \times \mathcal{I}^D & \rightarrow \quad d \ : \ d \subset \mathcal{I}^D \\ \langle M, I_{Train} \rangle & \mapsto \quad \{\langle x, y \rangle \in I_{Train} : \sum_{i=1}^{n} (\mathbf{1}_{\mathcal{X}_{s_i}}(x)) \geq t_{min}\}. \end{cases}$$

While equation (5.16) already characterises the subset of critical instances of the respective training set, i.e., $LP_{prop} \subset I_{Train}$, thus meeting the requirements of the `propExpand` function above, the misclassification count across the repeated cross-validations

$$c_{err}(lp) = \sum_{i=1}^{n} (\mathbf{1}_{\mathcal{X}_{s_i}}(x)) : lp = \langle x, y \rangle \in LP_{prop}$$

additionally allows for a total ordering of the elements $LP_{prop}$ based on the natural order $\leq$ of the $c_{err}(lp) \in \mathbb{N}^+$. Let $\leq_{err}$ denote that derived order, the computation of critical instances yields the ordered set $\mathcal{P}(LP_{prop}, \leq_{err})$.

Based hereupon, a simple procedure then involves the adoption of critical instances as a new learning problem, guided by the ordering that implies a degree of criticality. As a consequence, the query structure that is adopted in the multiagent interactive knowledge adaptation methodology is similar to those queries that are typically used in active learning scenarios (cf. Section 4.2.1).

As shown in Section 5.4.2, advice acquisition begins with the communication of those critical instances. They are specified as lesson contents in their un-labeled form. When the interaction with the advisors unfolds, and they classify the addressed instances in accordance with the withheld true class, a second stage of the interaction can be initiated. Therein, the focus of a subsequent query is the procurement of advice that is designed to replicate this classification choice on the advisee side.

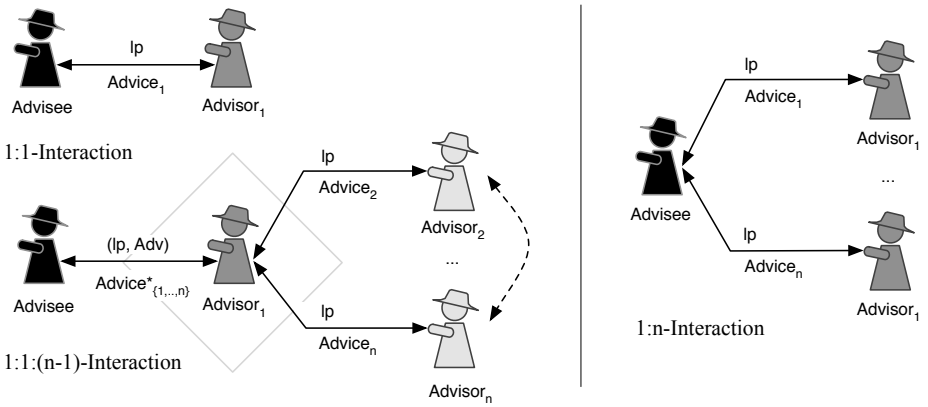



**Figure 5.9:** Conceivable interaction patterns for advice acquisition in the context of model adaptation episodes.

## 5.4.2 Structuring Agent Interaction for Advice Acquisition

In the modelling of multiagent interactive adaptation of classification models, as presented in the preceding Section 5.3.1, the acquisition of learning advice from an advisor panel *Advisors* is critical for the expansion of the momentary search state. To that end, the function `checkExpand` : $LP_{prop} \rightarrow \mathbb{B} \times LearningAdvice$ (see Equation (5.7) on page 106) stated that two pieces of information need to be acquired. First, whether or not the advisor panel is able to procure learning advice for the learning problem at hand (i.e., `canAdvise`$(Advisors, lp)$). In case of a positive feedback, the actual advice must be acquired (i.e, `getAdvice`$(Advisors, lp)$). The interaction with the advisor panel thereby ought to follow a specific course of events. For multiagent systems, it has been shown in Section 2.4.2, that interaction protocols are the customary means to model the process of knowledge acquisition at hand. They flesh out, and hence complement, the function specification introduced thus far.

Section 5.3.2 outlined how an advisee agent obtains information about advisors suitable for its classification task (cf. Algorithm 2). The advisee is then free to implement a custom policy with regard to the partitioning of this full advisor set *Advisors* into sub sets $Adv_i \in \mathcal{P}(Advisors)$ for consecutive adaptation episodes. Depending on the composition of the advisor panel, the interaction throughout the adaptation episode can fall into several interaction patterns, as sketched in Figure 5.9.

**Single-Tier 1:1 Interaction** As a first option, the advisee can choose a single dedicated advisor for exclusive interaction such that $\mid Adv_i \mid = 1$. Consequently, for each enquired learning problem $lp \in LP_{prop}$, the advisee can in the best case expect to receive a single piece of advice. If the advisor choses not to advise on the specified problem, the outcome may also be a failure of the exploration attempt.

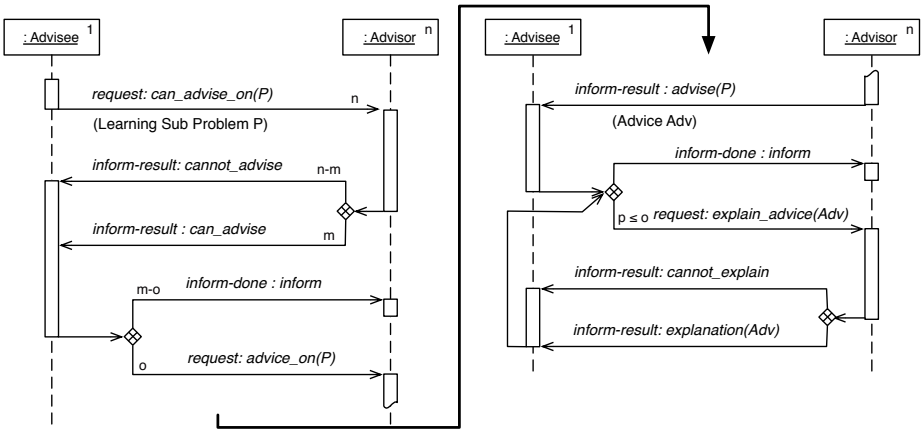



**Figure 5.10:** Protocol for single-tier 1:n interaction for advice acquisition.

**Single-Tier 1:n Interaction** To broaden advice acquisition for the learning subproblem, and at the same time reduce to the failure potential of the transfer episode, the advisee can chose place the same advisory request with a larger number of advisors, i.e. $\mid Adv_i \mid > 1$. As a result of this interaction pattern, the advisee can expect to receive multiple independently compiled pieces of advice. It lies within the responsibility of the advisee to consolidate the received advice pool as part of the advice integration which is done in state expansion (cf. Equation (5.8) on page 108).

**Two-Tier 1:1:n Interaction** The two interaction patterns discussed so far are single-tier interactions in which the advisee interacts directly with all advisors involved in the transfer episode. An alternative is a two-tier interaction which again involves a group of advisors. In contrast to the preceding interaction patterns, the advisors appear as a *holon*[2] with a single advisor acting as *holon head*.

The advisory holon is created dynamically upon request by the advisee. Besides the learning subproblem that constitutes the topic of the interaction, the advisee also communicates a set of additional advisors to an initial advisor acting as holon head. The initial advisor uses this information about the additional advisors to relay the learning problem and collects pieces of advice for further processing. Specifically, this interaction pattern can be applied when an advisor is willing to externalise the consolidation of the advice pool to a trusted external party. The interaction among the advisors can itself follow different interaction patterns depending on the preferred method of advice integration.

**A Single-Tier one-to-many Interaction Protocol** The following presents an interaction protocol for the single-tier 1:1 and 1:n interaction pattern (see Figure 5.10).

The protocol consists of several stages. First, the advisor requests from all potential advisors, whether or not they consider themselves fit to provide advice for the learning subproblem (i.e., function `canAdvise`$(Adv_i, lp)$ in Equation (5.7)).

2 In the categorisation of holonic MAS by Fischer et al. (2003), the organisational form of an *moderated association* is adequate.

The advisee waits for the advisor responses. Based on this feedback, it can determine the sub set of advisors from which it can then request actual advice (i.e., function `getAdvice`$(Adv_i^-, lp)$ where $Adv_i^- \subseteq Adv_i$ and $\forall\, adv \in Adv_i^-$ : `canAdvise`$(adv, lp) = \textsc{True}$).

The remaining advisors receive requests to advise. The interaction protocol in Figure 5.10 envisages that the earlier consent to advise enforces that each advisor actually provides individual advice and does not back out at this stage of the conversation. In case the communicated advice proves to be comprehensible for the advisee, the interaction with the respective advisor ends here. If, however, the advisee cannot comprehend any piece of advice directly, the interaction protocol provides for a continuation of the conversation to (repeatedly) request additional information to further understanding.

If the possibility for the advisee as protocol initiator to pose additional queries to advisors to further comprehension of the respectively procured learning advice is not factored in, the protocol in Figure 5.10 can be realised by means of the Contract Net protocol which was originally introduced in (Smith 1977) and later turned into a standard by the FIPA (Foundation for Intelligent Physical Agents 2002*c*).

### 5.4.3 Model-based Learning Advice

Section 5.3 which detailed the control mechanism for multiagent interactive adaptation and Section 5.4.2 which proposed an interaction protocol which is used in the process of advice acquisition, both used the concept of '*learning advice*' or simply '*advice*' colloquially as *some piece of information which is provided as a form of local domain-specific background knowledge that relates to a specific critical training instance*.

The characterisation as learning advice reflects the intended purpose of such information following its acquisition by the agent that assumes the advisee role in the adaptation process. Acquired learning advice is to provide a surplus value in inducing a concept description which mitigates a learning deficiency that is addressed during adaptation, compared to the situation without access to background knowledge.

#### Definition of Learning Advice

Section 5.4.1 discussed the identification of learning problems and showed that weaknesses in an induced conceptualisation, i.e., a classification model $M \in \mathcal{H}^D$, are associated with critical instances that were characterised by a higher-than average misclassification rate. If such critical instances constitute the subject matter for an advice acquisition interaction protocol, the sought learning advice needs to refer to these pre-set lesson contents. Beyond that, the specific type of learning advice furnished by an advisor needs to be chosen.

In Section 4.2.1 on active learning methods, and Section 4.2.2 on argument based machine learning, several types of advice have been used. Possible schemes that have been proposed in the literature comprise:

a) *Instance-based advice* in the form of new labeled/sampled instances,
b) Advice in the form of *global background knowledge* for the respective learning problem,
c) *Argument advice*, which constitutes a instance-specific rationale or justification for the correct classification of specific instances.

This thesis adopts a definition of *argument advice* that is consistent with the definition of '*arguments*' as proposed in Možina et al. (2007, p. 925).

The following formal definitions thus refer to this important groundwork. With respect to the fundamental distinction of two argument types, the following definition applies.

**Definition 5.13** (Supporting Argument)

*Let* $\langle \mathbf{x}, y \rangle \in \mathcal{I}^D$ *constitute an instance from a data set. It is characterised by the feature vector* $\mathbf{x} \in \mathcal{X}^D$ *and the concept value* $y \in Y^D$*. A supporting argument is then defined as*

$$arg^+ := \langle y, r_1 \wedge \ldots \wedge r_n \rangle \ : \ n \geq 1 \tag{5.17}$$

*which is interpreted as* $(C = y$ ***because*** $r_i \wedge \ldots \wedge r_n)$ *for the instance described by* $\mathbf{x}$*.*

*The* $r_i$ *are called* reasons*. They take one of five possible forms of selectors which refer to features* $F_j \in \mathcal{F}_{nom} \cup \mathcal{F}_{num}$ *that constitute dimensions of the feature space* $\mathcal{X}^D$*:*

*a)* $r_i = \langle F_j, =, v \rangle \ : \ v \in Dom(F_j)$, $v = \mathbf{x}[j]$*: The value* $f_k$ *of the feature* $F_j$ *is a reason why the instance* $\langle \mathbf{x}, y \rangle$ *belongs to the given class. This form of selector is the only allowed form for nominal, i.e., discrete, features.*

*b)* $r_i = \langle F_j, >, t \rangle \ : \ t \in Dom(F_j)$ *(or* $r_i = \langle F_j, \geq, t \rangle$*): The fact that the value of the feature* $F_j$ *for the given instance is greater than (greater or equal to) the threshold value* $f_k$ *is the reason for the given class.*

*c)* $r_i = \langle F_j, <, t \rangle \ : \ t \in Dom(F_j)$ *(or* $r_j = \langle F_j, leq, t \rangle$*): This is the opposite to the previous case.*

*d)* $r_i = \langle F_j, >, * \rangle$ *(or* $r_i = \langle F_j, \geq, * \rangle$*): This item can be understood as a relaxation of the third form of reason. It states that the fact that the value of the feature* $F_j$ *is 'high' or 'high enough' is a reason for the instance for fall into the given class. In other words, the existence of a threshold value is assumed but deliberately left unspecified a part of the inclusive advice.*

*e)* $r_i = \langle F_j, <, * \rangle$ *(or* $r_i = \langle F_j, \leq, * \rangle$*): This is the opposite to the previous case.*

*The set of valid supporting arguments shall be denoted as* $Arg^+$*.*

**Definition 5.14** (Inverse Supporting Argument)

*Let* $\langle \mathbf{x}, y \rangle$ *constitute an instance from a data set which is characterized by the feature vector* $\mathbf{x} \in \mathcal{X}^D$ *and the concept value* $y \in Y^D$*. An inverse supporting*

*argument is defined as*

$$arg^{-} := \langle\, y, r_1 \wedge \ldots \wedge r_n \,\rangle \;:\; n \geq 1 \tag{5.18}$$

*which is interpreted as* $(C = y$ ***despite*** $r_i \wedge \ldots \wedge r_n)$ *for the instance described by* $\mathbf{x}$.

*As with supporting arguments, each reason* $r_i$ *can take one of the five possible form of selectors which refer to features* $F_j$ *that constitute a dimension of the feature space* $\mathcal{X}^D$. *The respective semantics is thereby polar to the inclusive case. The set of valid inverse supporting arguments shall be denoted as* $Arg^{-}$.

It should be noted that Možina et al. (2007, p. 925ff) originally denoted supporting arguments as *positive arguments* and inverse supporting arguments as *negative arguments*. The naming deviation has been introduced to clarify the intended semantics of negative arguments. Such arguments convey that the reasons within the argument should not play a role for a positive classification of the given instance $\langle\, \mathbf{x}, y \,\rangle$.

With these definitions as prerequisites, it is now possible to provide a definition of a complete piece of argument advice as follows.

**Definition 5.15** (Argument Advice)

*Let* $Arg^{+}$ *denotes the set of supporting arguments.* $Arg^{-}$ *denotes the set of inverse supporting arguments. A piece of argument advice* $Adv^{arg}$ *is defined as a set of arguments with* $|Adv^{arg}| \geq 0$. *For each argument* $arg_i \in Adv^{arg}$, *it holds that* $arg_i \in Arg^{+} \cup Arg^{-}$.

### 5.4.4 Preparation of Learning Advice

As highlighted in Section 4.2.2, the original application scenario for argument based machine learning presumes that argument advice conforming to Definition 5.15 is procured from a human domain expert. In advice procurement, the expert explicates his/her expertise with respect to a specific case. While this mental process is of complex nature, its implementation is not part of the examined experimental setting. It is simply assumed that due to the comprehensible composition of argument advice which also explicitly accounts for vague formulations, the domain expert can communicate on a machine-interpretable level.

The transition from a machine-to-human learning setting to a machine-to-machine or specifically agent-to-agent setting entails that the aforementioned advisor capability is emulated by an intelligent agent. Hence, the agent needs to provide a function which maps from its own classification model and a specific critical instance onto a piece of argument advice.

$$\texttt{computeAdvice} : \begin{cases} \mathcal{H}^D \times \mathcal{X}^D \times \mathcal{Y}^D & \rightarrow \quad Advice \\ \langle\, M, \mathbf{x}, y \,\rangle & \mapsto \quad Adv^{arg} \end{cases} \tag{5.19}$$

The scope and intricacy of this function depends on the type of classification

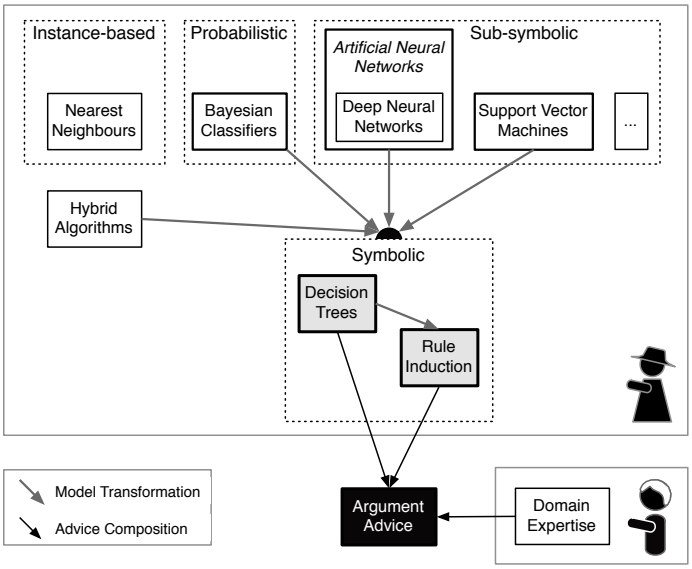



**Figure 5.11:** Schematic overview of feasible transformations of different types knowledge representations that are used for classification purposes. The graphics also highlights the decomposition of the task to compute argument advice.

model that is employed by an advisor. Section 4.1 highlighted the landscape of machine learning branches. In particular, as sketched in Figure 4.2, it has presented the main families of classification models that ensue with a partition based on the underlying knowledge representation. Specifically, it highlighted the distinction between symbolic representations $\mathcal{H}^D_{sym} \subset \mathcal{H}^D$, probabilistic representations $\mathcal{H}^D_{prob} \subset \mathcal{H}^{\mathcal{D}}$, and sub-symbolic representations $\mathcal{H}^D_{sub} \subset \mathcal{H}^D$, focusing on pure representations which are induced by eager learning schemes.

In the characterisation of the advisor knowledge management role in Section 5.2.2, it has been argued that this function is a critical constituent for transferring the principal mechanisms of argument based machine learning to multiagent systems.

If one renders the design decision to adopt the structure of argument advice as introduced in the preceding section, thereby retaining the opportunity to enable a heterogeneous advisory board formed by intelligent software agents and human domain experts, the syntactical structure of argument advice becomes the common denominator for advice induction from the aforementioned types of knowledge representations.

Symbolic models such as induced *propositional rule sets* as in CN2 (Clark & Niblett 1989, Clark & Boswell 1991), RIPPER (Cohen 1995), PRISM(Cendrowska 1987) or the AQ (Wojtusiak et al. 2006) family of algorithms (see Fürnkranz (1999) for a survey) one the one hand, or *decision trees* as in C4.5 (Quinlan 1993) on the other already possess comprehensible knowledge representations (Freitas 2014) with strong similarities to argument advice. Also, Quinlan (Quinlan 1987*a*,*b*) and other researchers have also proposed methods for the extraction of preferably compact propositional rule sets from trained decision trees, for the most part in an attempt to achieve an increase in comprehensibility.

Probabilistic representations, such as Bayesian networks, and in particular sub-

symbolic representations, as exemplified by support vector machines (Cristianini et al. 2000) and artificial neural networks (Rojas 2013), amongst them also the deep neural network architectures employed in Deep Learning (Goodfellow et al. 2016), have an internal structure which demands a higher investment to transform such models into symbolic representations.

Section 4.1.4 has discussed paradigmatic examples from the substantial body of research on methods to derive explanations from trained black-box models based on sub-symbolic representations or ensembles (Guidotti et al. 2018) which includes the extraction of comprehensible symbolic representations.

As a consequence of the principal feasibility of the transformation of heterogeneous classification models into a comprehensible, symbolic form as sketched in Figure 5.11, a decomposition of the task to prepare argument advice is introduced. For non-symbolic models, it results in a two-tiered integration approach whose first tier uses a model transformation function

$$\texttt{transformModel} : \begin{cases} \mathcal{H}^D \setminus \mathcal{H}^D_{sym} & \rightarrow & \mathcal{H}^D_{sym} \\ M_{orig} & \mapsto & M_{extr} \end{cases} \tag{5.20}$$

Given such a transformation function, whose model-specific appropriation is the goal of the aforementioned research efforts, Equation 5.19 can be reduced to the following form after preprocessing that provides a desired model transformation.

$$\texttt{computeAdvice} : \begin{cases} \mathcal{H}^D_{sym} \times \mathcal{X}^D \times \mathcal{Y}^D & \rightarrow & \mathit{Advice} \\ \langle\, M, \mathbf{x}, y \,\rangle & \mapsto & \mathit{Adv}^{arg} \end{cases} \tag{5.21}$$

While the effectiveness of such a divide-and-conquer approach to the task of advice provision needs to be examined – especially with respect to losses in model content that result from the transformation, measured against the benefits of an incorporation of knowledge from heterogeneous origins into the advisory process –, it allows to start and develop an extensible methodology with a focus on well-understood and comprehensible symbolic models.

The following paragraphs describe a heuristic approach as shown in Algorithm 3. It assumes an advisor with a propositional rule set as its classification model. As sketched above and amplified in Section 4.1, a large body of propositional rule induction systems has been developed and can be employed.

Algorithm 3 is characterised as a direct and selection-based approach to the problem of extracting arguments from an existing rule set. It works under the assumption that the selection from a subset of rules in a propositional rule set, where these rules cover a given critical instance according to (1) *rule quality* as primary selection criterion and (2) *rule complexity* as secondary criterion, already procures the basis for feasible advice. The rationale here is that the rules themselves are already understood as explanations for class assignment given their covering.

The *rule quality* can be evaluated in different ways. Possible techniques discussed in the literature comprise amongst others *Laplace's Rule of Succession*, *Weighted Relative Accuracy* (see (Lavrač et al. 1999)) or *Extreme Value Correction* (see (Možina et al. 2007, p. 928)). The evaluation function thereby typically

**Algorithm 3** Heuristic function for procurement of argument advice via a rule selection procedure.

**Input:** $\mathcal{R}$ : Rule set induced for the classification problem
$\langle \mathbf{x}, y \rangle$ : Critical instance for which advice is sought
**Output:** $adv_{arg}$ : A piece of argument advice, initially the empty set

```
1: function computeAdviceSingle(R, ⟨x, y⟩)
2:     for (rule_i : R where covers(rule_i, ⟨x, y⟩) = True) do % Source Rule Selection
3:         if (rule_best = None) then
4:             rule_best ← rule_i
5:         else if (qual(rule_i) > qual(rule_best)) then
6:             rule_best ← rule_i
7:         else if (qual(rule_i) = qual(rule_best) and compl(rule_i) < compl(rule_best)) then
8:             rule_best ← rule_i
9:     if rule_best ≠ None then % Argument Composition
10:        arg ← ∅
11:        for (sel_i : selectors(rule_best)) do
12:            if (type(sel_i) = Nominal) then
13:                arg ← arg ∪ (sel_i +' =' +value(x, sel_i))
14:            else if (type(sel_i) = Numeric) then
15:                arg ← arg ∪ (sel_i + sign(sel_i) + value(x, sel_i))
16:        adv_arg ← adv_arg ∪ arg
17:    return adv_arg
```

corresponds to the function also used in the induction process of separate-and-conquer rule learners such as CN2.

The *complexity* of a propositional rule is typically measured by the number of selectors that constitute its premise.

Especially in cases where multiple covering rules exist, an investigation on whether the consideration of all rules above a quality threshold, their consolidation, and further processing designed to reduce the complexity of arguments leads to a significant improvement with regard to the integration of argument advice as outlined in the following section is worthwhile. The next section develops algorithms for advice consolidation.

### 5.4.5 Generalisation of Learning Advice

The preceding sections laid the foundation for the acquisition of learning advice in the context of model adaptation episodes. Specifically, Section 5.3.1 provided the context for the process of knowledge acquisition by modelling the adaptation of classification models as an online search problem. Section 5.3.2 showed that

```
[0] Advisor.01 | [0101000100000000000100] |
    {aspect<=334.0,elevation<=2314.0,horiz_dist_to_fire_points<=808.0,
     horiz_dist_to_hydrology<=0.0}
[1] Advisor.02 | [0100000000010000000100] |
    {elevation<=2216.0,horiz_dist_to_fire_points<=912.0,
     horiz_dist_to_roadways<=1298.0}
[2] Advisor.02 | [0100000101001000000000] |
    {elevation<=2242.0,hillshade_9am>192.0,horiz_dist_to_hydrology<=30.0,
     vert_dist_to_hydrology<=11.0}
[3] Advisor.03 | [0111000100000000000000] |
    {aspect<=93.0,aspect>16.0,elevation<=2311.0,horiz_dist_to_hydrology<=0.0}
[4] Advisor.03 | [0110000001000000000100] |
    {aspect>16.0,elevation<=2309.0,horiz_dist_to_fire_points<=499.0,
     vert_dist_to_hydrology<=3.0}
[5] Advisor.03 | [0100000000011000000000] |
    {elevation<=2241.0,hillshade_9am>214.0,horiz_dist_to_roadways<=1357.0}
```

**Figure 5.12:** An example for multiple rules procured by the selected advisor panel which cover a single critical instance. The example is taken from an experiment with the UCI land cover type domain (Blackard et al. 1999).

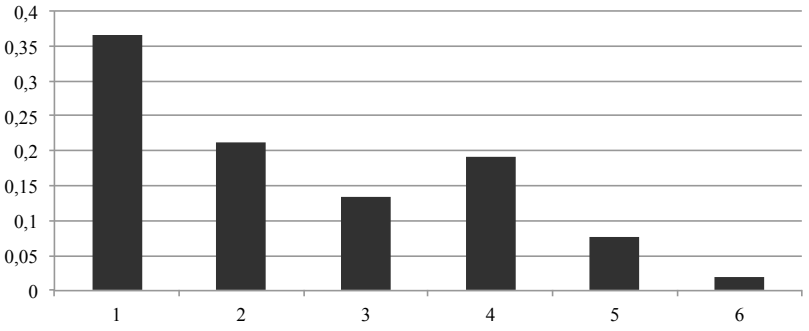



**Figure 5.13:** Normalised distribution of the number of rules covering addressed critical instances in an experiment with three CN2-based advisors, executed on the UCI cover type data set (see Section 7.3)

an advisee can adopt different policies to compose a panel of advisors for an adaptation episode. Specifically, it is possible to enlist a – potentially large – group of advisors with heterogeneous individual models for the desired learning task. Section 5.4.2 contributed an interaction protocol that structures advice acquisition from multiple advisors.

As result of a successful round of advice acquisition for a given learning problem, the advisee can, in the best case, obtain an advice pool whose size corresponds to the number of agents in the advisor panel. This assertion thereby refers to a situation as exemplified by Algorithm 3 in the preceding section, where process of advice composition on the advisor side already let to the procurement of a piece of argument advice with a single included argument. However, it should be noted that there is no general constraint to a single argument. Consider, for instance, that instead of selecting a single best rule to be interpreted as a positive argument (see Algorithm 3 on page 124), the advisor decides to employ a more liberal strategy where it turns the best $n$ firing rules into respective arguments. In such a case, the number of arguments which are procured by the different agents in the advisor panel can exceed the panel size.

This situation is exemplified in Figures 5.12 and 5.13. Specifically, Figure 5.12 offers a paradigmatic example where multiple firing rules for the critical instance addressed in a learning problem lead to multiple feasible arguments which are returned as part of the composed argument advice. Figure 5.13 shows a normalised histogram of the distribution of the amount of acquired arguments for single learning problems. This histogram, which has been generated in a supplementary experiment that spanned ten repetitions of the same adaptation experiment with a panel of three advisor agents, indicates that for a significant fraction of cases, the advisee can acquire a notable set of arguments. This holds even for as few as three consulted advisors.

This situation affords the advisee with a data basis to post-process the acquired arguments for a learning problem such that accordances amongst the arguments can be carved out while idiosyncrasies of particular arguments which are not backed by further sufficiently similar arguments can be inhibited. The goal of the type of pre-processing which is introduced in the following is an aggregation and gentle generalisation of advisor-procured arguments before the consolidated learning advice is passed on for the expansion of the current search state in the context of a model adaptation episode (see Equation (5.8) on page 108).

### Binarisation of Arguments

The aggregation of arguments presupposes a representation which allows for the determination of their degree of similarity. To that end, a bit vector encoding for arguments is introduced. Section 5.3 formally introduced basic concepts which are used here. The classification task for which an advisee seeks to improve its momentary model has been introduced as a tuple $Task_{Cl} := \langle\, Domain, Y^D, \texttt{classify} \,\rangle$ (see Definition 5.2) where the data domain $Domain := \langle\, \mathcal{X}^D, \mathbf{P}(X) \,\rangle$ is characterized by the chosen task-specific feature space $\mathcal{X}^D$ (see Definition 5.1). The feature space for a classification task with both numeric and nominal features is given as

$$\mathcal{X}^D := F_1 \times \ldots \times F_i \times F_{i+1} \times \ldots \times F_n \;:\; 1 \leq i \leq n.$$

$$\text{where } \{F_1, \ldots, F_i\} \in \mathcal{P}(\mathcal{F}_{nom}), \{F_{i+1}, \ldots, F_n\} \in \mathcal{P}(\mathcal{F}_{num})$$

Both inclusive arguments $arg^+$ and exclusive arguments $arg^-$ as defined in the Definition 5.13 on page 120 and Definition 5.14 on page 121 use a subset of the features defined in $\mathcal{X}^D$.

For each $n$-dimensional feature space, a corresponding binarised representation can be constructed. The construction rule for this bit vector denoted as $\mathbf{b} \in \mathbb{B}^m$ is as follows, assuming a fixed ordering $\succ_{\mathcal{F}}$ of the features $F_1, \ldots, F_n \in \mathcal{F}_{nom} \cup \mathcal{F}_{num}$ as preset by the above Cartesian product or a lexicographical ordering.

a) Each nominal feature $F_j \in \{F_1, \ldots, F_i\}$ is represented in the bit vector $\mathbf{b}$ by a single bit which indicates, whether the respective feature $F_j$ appears in a reason $r_i = \langle\, F_j, =, v \,\rangle$.
b) Each numerical feature $F_k \in \{F_{i+1}, \ldots, F_n\}$ is represented in the bit vector by two bits. The first bit indicates whether the respective feature $F_j$ appears in a reason $r_i = \langle\, F_j, \{<|\leq\}, \{t \mid *\} \,\rangle$, i.e., whether an upper threshold for

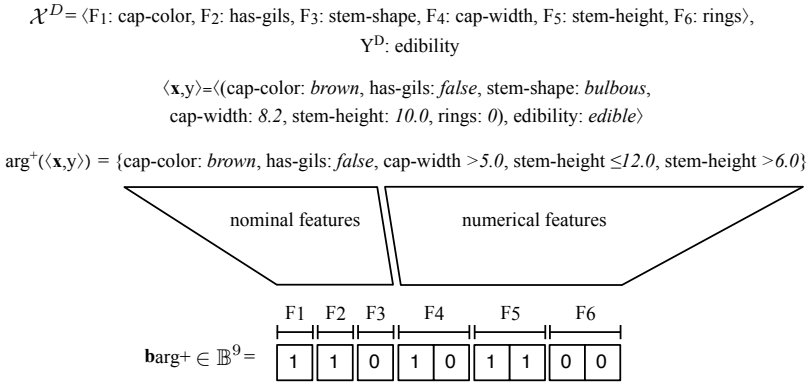



**Figure 5.14:** A paradigmatic example for the binarisation of a positive argument $arg^+$ into an according bit vector $\mathbf{b}$.

the nominal feature is implied.
The second bit covers the complimentary case. It hence indicates where the feature $F_j$ appears in a reason $r_j = \langle\, F_j, \{>|\geq\}, \{t \mid *\}\,\rangle$.

The introduced bit vector by itself represents a coarse representation of the original argument. An example for the coding scheme is shown above in Figure 5.14. The number of bits that are required to encode any argument is $|\mathbf{b}| = m = i+2\cdot(n-i)$.

Based in the binarisation presented above, two additional concepts are defined which are required for an assessment of the similarity of two arguments.

**Definition 5.16** (Argument Overlap)

*Let $arg_i, arg_j$ denote two inclusive (exclusive) arguments and let $\mathbf{b}_i, \mathbf{b}_j$ denote their respective binarised form, according to the construction rules given above. Then, the overlap of these two arguments is calculated as*

$$\texttt{overlap}: \begin{cases} \mathbb{B}^m \times \mathbb{B}^m & \rightarrow & \mathbb{N}_o^+ \\ \langle\, \mathbf{b}_i, \mathbf{b}_j \,\rangle & \mapsto & |\, \mathbf{b}_{int} \,|: \; \mathbf{b}_{int}[k] = \textsc{True} \\ & & \quad \textit{iff}\ \texttt{and}(\mathbf{b}_i[k], \mathbf{b}_j[k]) = \textsc{True}. \end{cases} \tag{5.22}$$

**Definition 5.17** (Argument Disparity)

*Let $arg_i, arg_j$ denote two inclusive (exclusive) arguments and let $\mathbf{b}_i, \mathbf{b}_j$ denote their respective binarised form. Then the disparity of these two arguments is calculated as*

$$\texttt{disparity}: \begin{cases} \mathbb{B}^m \times \mathbb{B}^m & \rightarrow & \mathbb{N}_o^+ \\ \langle\, \mathbf{b}_i, \mathbf{b}_j \,\rangle & \mapsto & |\, \mathbf{b}_{int} \,|: \; \mathbf{b}_{int}[k] = \textsc{True} \\ & & \quad \textit{iff}\ \texttt{xor}(\mathbf{b}_i[k], \mathbf{b}_j[k]) = \textsc{True}. \end{cases} \tag{5.23}$$

Argument overlap and disparity as given in these definitions are used subsequently to cluster an acquired argument pool into sets of similar arguments which can then be considered for generalisation.

### Clustering of Arguments

For the clustering of arguments in the presented application scenario, a suitable clustering method had to be chosen. To that end, the following considerations apply. First, Figure 5.13 has presented preliminary empirical results which state that the total amount of points $\mathbf{b}_i \in \mathbb{B}^D$, i.e., arguments, in the input data is low compared to typical clustering scenarios. Second, the histogram in Figure 5.13 also shows that the size of the input data also varies significantly across learning problems that are addressed in adaptation episodes. Third, the distribution of the $\mathbf{b}_i$ depends on the respective composition of the advisor panel for adaptation episodes and the characteristics of the classification models that are consulted to compile argument learning advice.

Consequently, centroid-based clustering such as k-means and related approaches which generally seek to partition the input data into a pre-determined, fixed number of clusters has not been considered. However the notion of clusters as areas in d-dimensional space where objects reside in relative proximity to each other, and which are divided by sparsely populated areas, which is adopted by density-based clustering approaches (Kriegel et al. 2011), was affirmed by experiments. It is therefore obvious to consider density-based clustering for arguments using the popular DBSCAN algorithm (Ester et al. 1996)[3].

An analysis of DBSCAN characteristics with respect to the classification task presented in the context of advice consolidation leads to mixed results. Since the scarceness of the input data set for argument clustering requires a small parameter $t_{min}$, the produced clusters are prone to exhibit the *single-link effect* which also results from the transitive extension of the `expandCluster`() function. This effect can be best conceived by a cluster in two-dimensional space. Its shape is formed such that it features an elongated tube-like shape. When clustering arguments, the effect translates to clusters whose elements have only a very basic commonality with respect to the set dimensions in their binary representation. This in turn is not a desirable property with regard to the sought-for generalisation of based on the cluster members. In fact, in order to guard against overly aggressive syntactic generalisation, the input clusters should exhibit a spatial compactness.

Another undesirable property of DBSCAN is given in Lemma A.2 on page 256. Since the clusters of arguments that are to be constructed constitute but the point of origin for an argument consolidation, a partial overlap of clusters is not at all considered a problem. It rather constitutes an opportunity, as the data basis for consolidation is broadened.

As a consequence, a non-transitive clustering procedure which explicitly ensures the compactness of clusters and allows for partial cluster overlap is proposed in Algorithm 4 on the next page.

Besides the full set of arguments that have been acquired as argument learning advice for a particular learning problem, the algorithm accepts a pair of parameters which control the clustering process. The first parameter $\epsilon$ specifies the upper bound for the argument bit vector dissimilarity. The second parameter $t_{int}$ specifies a lower bound on the cluster intersection. Initially, all input arguments are fed into an open list. This data structure contains all arguments which are

3 The formal basis for density-based clusters is available at Section A.1 in the appendix.

**Algorithm 4** Clustering procedure for binarised arguments.

**Input:** $\epsilon \in \mathbb{R} \ : \ 0 < \epsilon \leq 1$ : Upper bound for argument bit vector dissimilarity
$t_{int} \in \mathbb{R} \ : \ 0 < \epsilon \leq 1$ : Lower bound for cluster intersection
$Arguments$ : An input set of bit vector representations of acquired arguments, i.e., $Arguments = \{\mathbf{b} \mid \mathbf{b} \in \mathbb{B}^D\}$.
**Output:** $Clusters = \{Cl \mid Cl \in \mathcal{P}(Args)\}$ : Set of argument clusters, initially empty

```
 1: function doCluster(Arguments, ϵ, t_int)
 2:     Clusters ← ∅, openList ← Arguments
 3:     while (openList ≠ ∅) do
 4:         Cl ← ∅, seed ← next(openList)
 5:         openList ← openList \ {seed}
 6:         Cl ← Cl ∪ {seed}
 7:         for (neighbor in regionQuery(seed, Arguments, ϵ, t_int)) do
 8:             openList ← openList \ {neighbor}
 9:             Cl ← Cl ∪ {neighbor}
10:         Clusters ← Clusters ∪ Cl
11:     return Clusters

12: function regionQuery(seed, Arguments, ϵ, t_int)    % Region query is not
    transitive
13:     neighbors ← ∅
14:     for (arg : Arguments \ {seed}) do
15:         if (dist(seed, arg) < ϵ and  % see Equation (5.24)
16:             neighborhoodComp(seed, neighbors, arg, t_int) = True) then
17:             neighbors ← neighbors ∪ {argument}
18:     return neighbors

19: function neighborhoodComp(seed, neighbors, neighborCand, t_int)
20:     intersect, union ← seed
21:     for (arg : neighbors ∪ {neighborCand}) do
22:         intersection ← and(intersection, arg)
23:         union ← or(union, arg)
24:     return (|intersection| / |union| ≥ t_int)
```

still eligible as seeds for a further argument cluster. While the open list is not yet empty, a single argument is drawn as seed for a new cluster and subsequently removed from the list. Then, the subordinate function `regionQuery` (Alg. 4, line 12) is invoked with the seed, the full set of input arguments, and the pair $\langle\, \epsilon, t_{int} \,\rangle$ as parameters. Similar to DBSCAN, the region query determines an epsilon neighbourhood of similar arguments. However, the function is not transitive. It uses the definitions of argument disparity (see Definition 5.17) and argument

overlap (see Definition 5.17) for the distance function

$$\texttt{dist} : \begin{cases} \mathbb{B}^m \times \mathbb{B}^m & \rightarrow \quad d \ : \ d \in \mathbb{R}, 0 \leq d \leq 1 \\ \langle\, \mathbf{b}_i, \mathbf{b}_j \,\rangle & \mapsto \quad \dfrac{\texttt{disparity}(\mathbf{b}_i, \mathbf{b}_j)}{(\texttt{disparity}(\mathbf{b}_i, \mathbf{b}_j) + \texttt{overlap}(\mathbf{b}_i, \mathbf{b}_j))}. \end{cases} \tag{5.24}$$

The function iterates over all provided arguments and determines, whether they lie within the neighbourhood of the current seed argument *seed* with respect to the parameter $\epsilon$. If an argument $arg$ satisfies $\texttt{dist}(seed, arg) < \epsilon$, it is considered a candidate neighbour. However, it may still be rejected as proper neighbour on the grounds that an admission would violate cluster compactness. This is determined by the function `neighborhoodComp` (Alg. 4, line 19). It constructs a temporary extended cluster containing all proper neighbours determined thus far and the neighbour candidate. Subsequently both the intersection and union bit vector are constructed for all cluster elements. A neighbour candidate is admitted if the ratio $|intersection| \,/\, |union| \geq t_{int}$. Hence, the compactness of the constructed clusters can be controlled effectively.

When a new cluster has been constructed, its constituents are removed from the open list. This ensures that further clusters begin with seed arguments which lie outside the boundaries of all thus far constructed clusters. However, since the complete set of arguments is passed to the region query as parameter, the possibility is retained for clusters to grow such they overlap with previous clusters. albeit only with non-seed arguments.

### Syntactic Generalisation of Argument Clusters

Clusters of arguments which are constructed with the help of Algorithm 4 on the preceding page constitute sets of arguments which are similar with respect to $\langle\, \epsilon, t_{int} \,\rangle$ in their propositions for the addressed learning problem. The generalisation of these original arguments into a consolidated single argument seeks to pronounce the cluster quintessence.

The proposed approach is based on syntactic similarity for arguments. It is specified in Algorithm 5. The algorithm accepts as input a preset cluster of similar arguments in their respective binarised representations and the matching set of original arguments. It also accepts two additional parameters that shape the generalisation process. The first parameter $t_{maj}$ represents a lower bound for the fraction of cluster elements that need to support a particular type of reason in order to retain that reason throughout generalisation. The boolean flag *shadow* decides whether concrete thresholds should be retained for reasons on numerical features.

The syntactic generalisation in Algorithm 4 distinguishes two phases. In the first phase, the argument bit representations are processed to compute a cluster agreement bit vector $\mathbf{b}_{join}$ whose set bits determine, which reasons are retained in the final generalised argument. The parameter $t_{maj}$ can be used to realise different policies in computing $\mathbf{b}_{join}$.

a) $t_{maj} = 1$ corresponds to a conservative policy where a reason needs to occur in each single cluster element to be retained.

**Algorithm 5** Procedure for the syntactic generalisation of a prepared cluster of argument bit vectors.

**Input:** $t_{maj} \in \mathbb{R} \ : \ 0 < t_{maj} \leq 1$ : Threshold for setting a bit in $\mathbf{b}_{join}$
$shadow \in \mathbb{B}$ : Flag which indicates whether concrete thresholds should be retained
$Cluster$ : An input set of bit vector representations of acquired arguments, i.e., $Cluster = \{\mathbf{b} \mid \mathbf{b} \in \mathbb{B}^D\}$
$Args$ : The acquired set of original arguments
**Output:** $\langle\, \mathbf{b}_{join}, generalizedReasons \,\rangle$ : A single generalized argument

1: **function** SYNTACTICGENERALIZATION($Cluster, t_{maj}, shadow, Args$)
2: $\mathbf{b}_{ref} \leftarrow$ `next`($Cluster$)
3: $bitAlloc \leftarrow$ `init`() : $bitAlloc \in (\mathbb{N}_0^+)^m$
4: **for** ($arg : Cluster$) **do** `% Determine argument selector distribution`
5: **for** ($i \leftarrow$ `nxtSetBit`($arg, 0$); `hasNextSetBit`($arg$); $i \leftarrow$ `nxtSetBit`($arg, i+1$)) **do**
6: $bitAlloc[\,i\,] \leftarrow bitAlloc[\,i\,] + 1$
7: $\mathbf{b}_{join} \leftarrow \mathbf{b}_{ref}$, `reset`($\mathbf{b}_{join}$)
8: **for** ($i = 0; i <$ `length`($bitAlloc$); $i \leftarrow i + 1$) **do** `% Determine cluster agreement`
9: **if** (($bitAlloc[\,i\,]$/`size`($Cluster$)) $> t_{maj}$) **then**
10: $\mathbf{b}_{join}[\,i\,] \leftarrow$ TRUE
11: $generalizedReasons \leftarrow \emptyset$
12: **for** ($i \leftarrow$ `nxtSetBit`($\mathbf{b}_{join}, 0$); `hasNextSetBit`($\mathbf{b}_{join}$); $i \leftarrow$ `nxtSetBit`($\mathbf{b}_{join}, i + 1$)) **do**
13: **if** (`encodesNumericFeature`($\mathbf{b}_{join}[\,i\,]$) = TRUE) **then**
14: $r_{gen} \leftarrow \langle *, *, * \rangle$
15: **if** (`encodesLowerThreshold`($\mathbf{b}_{join}[\,i\,]$) = TRUE) **then**
16: **for** ($argument : Cluster \ : \ argument[\,i\,]$ = TRUE) **do**
17: **if** ($r_{gen} \neq \langle *, *, * \rangle$) **then**
18: $r_{gen} \leftarrow$ `getReason`($argument, i, Args$), **continue**;
19: $r_{arg} \leftarrow$ `getReason`($argument, i, Args$)
20: **if** (($r_{arg}.value > r_{gen}.value$) **or**
21: ($r_{arg}.value = r_{gen}.value$) **and** $r_{arg}.sign =$ " > ") **then**
22: $r_{gen} \leftarrow r_{arg}$
23: **else if** (`encodesLowerThreshold`($\mathbf{b}_{join}[\,i\,]$) = TRUE) **then**
24: **for** ($argument : Cluster \ : \ argument[\,i\,]$ = TRUE) **do**
25: **if** ($r_{gen} \neq \langle *, *, * \rangle$) **then**
26: $r_{gen} \leftarrow$ `getReason`($argument, i, Args$), **continue**;
27: $r_{arg} \leftarrow$ `getReason`($argument, i, Args$)
28: **if** (($r_{arg}.value > r_{gen}.value$) **or**
29: ($r_{arg}.value = r_{gen}.value$) **and** $r_{arg}.sign =$ " < ") **then**
30: $r_{gen} \leftarrow r_{arg}$
31: **if** ($shadow$ = TRUE) **then**
32: $r_{gen}.value \leftarrow *$
33: $generalizedReasons \leftarrow \langle\, i, r_{gen} \,\rangle$
34: **return** $\langle\, \mathbf{b}_{join}, generalizedReasons \,\rangle$

b) $t_{maj} = 0.5$, by contrast, corresponds to a comparatively liberal policy where it is only required for a reason to appear in the majority of cluster elements.

In the second phase of the algorithm, the thresholds for reasons based on numerical features are computed in a conservative way. Specifically, the most restrictive upper and lower thresholds are carried over respectively, assuming that *shadow* = TRUE. Otherwise, thresholds are dropped completely and replaced by placeholders. It should be noted here, that the semantic of a reason that refers to a numeric features but forgoes specifying a specific threshold is to convey that the value of a certain feature being big (small) is important in itself but that concrete thresholds out to be determined self-sufficiently.

Algorithm 5 returns a pair $\langle \mathbf{b}_{join}, generalizedReasons \rangle$ which contains all necessary information to re-build a proper argument for inclusion in argument advice.

### 5.4.6 Internalisation of Learning Advice

The final essential building block in the presented methodology for multiagent interactive adaptation addresses the utilisation of acquired learning advice. In taking up the advice and using it as additional knowledge that allows for the computation of a model derived from the current state in the search space, actual options to leave the location are procured. Several approaches for the utilisation of advice are conceivable.

The first class is characterized by direct alteration of the source model, based on the contents of acquired advice. An alternative to approaches that build upon an iterative alteration of an original trained base model is put into practice in both active learning and argument based machine learning (see Section 4.2.1).

These methods have as a common denominator that they iteratively re-train the respective model with a growing pool of input data. The character of this growth thereby has a bearing on the learning schemes that can be utilised for re-learning. Specifically, active learning methods based on instance queries grow the number of instances used for training. As a consequence, the learning scheme does not need to be adapted. Argument based machine learning, by contrast, retains the initial number of learning instances over the course of a model adaptation process. Instead, it collects argument advice for few, well-chosen critical instances. This growing pool of arguments then must be accounted for in each re-learning step. This in turn necessitates a modification of the base form of learning scheme used by the advisee. To that end, Možina et al. (2006, 2007, 2010) have proposed ABCN2 as an argument based version of the well-known CN2 rule learner. Similarly, Napierala & Stefanowski (2010), have proposed an argument based generalisation of the MODLEM (Stefanowski 1998) algorithm.

It should be noted that both approaches to the acquisition and subsequent employment of additional information for model improvement are complements. The question of which to employ in a given situation depends on the feasibility to acquire either information.

This thesis focusses on the argumentation-based approach that necessitates the

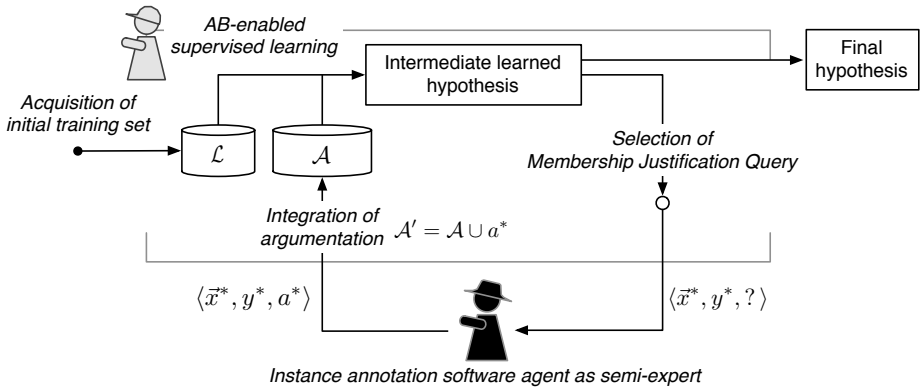



**Figure 5.15:** The active learning cycle for argument based machine learning applied in a multiagent environment, assuming a 1:1 interaction pattern (see Section 5.4.2).

integration of multiagent technology, agent-oriented knowledge management and machine learning, based on a scenario as described in Section 5.1 on page 88. Hence, ABCN2 is adopted in order to re-learn models upon the arrival of new argument advice, in order to compute potential successor states in the adaptation search process. Section A.2 in the appendix provides details on the ABCN2 algorithm, which refers to work by Možina et al. (2006, 2007).

## 5.5 Chapter Summary

This chapter has presented a holistic approach to the challenge of a peer-supported adaptation of the individual classification models of a learning agent. Following an initial investigation of scenarios and prerequisites, a full-fledged adaptation methodology has been developed on multiple consecutive and deepening levels of description. First, the identified challenge has been approached from a knowledge management point of view. As a result, a system of knowledge management roles has been proposed which includes both the essential roles for knowledge consumers and knowledge providers and additional roles to form knowledge networks dynamically within multiagent systems. Second, the process of model adaptation with a preset advisor panel in the context of knowledge adaptation episodes has been embraced from an artificial intelligence perspective. It has been modelled formally in terms of an online search problem. These searches then have been integrated into the framework of multiagent interactive adaptation which interleaves advisor acquisition and model adaptation episodes. One contribution of the concept is thereby the decomposition of the initial problem statement. This allowed for a subsequent modular treatment of important problem aspects such as the denomination of learning problems, the structuring of the interaction among advisee and advisors, as well as the advisor-side composition and advisee-based consolidation and utilisation of learning advice in the presented adaptation methodology. The latter modules thereby employed techniques from both distributed artificial intelligence, specifically multiagent coordination, and argument-based machine learning.

Summing up, the chapter presented an end-to-end concept for peer-supported model adaptation in multiagent systems, for which a complete implementation is subject of the following chapter. The problem decomposition introduced in this chapter is flexible with regard to future extension which is discussed further in Chapter 8.

Chapter 6

# Reference Implementation

Building on the conceptual design of interactive multiagent adaptation of classification models, this chapter provides details on the realised reference implementation for JADE multiagent systems.

To that end, Section 6.1 introduces a flexible role-based architecture for intelligent agents. This architectural principle is consequently applied in Section 6.3 for the implementation of those management functions that are required for the formation and cultivation of dynamic knowledge networks for peer-supported classification model adaptation among knowledge based learning agents. Before focussing on these knowledge management functions, Section 6.2 first introduces the learning infrastructure which procures the fundamental building blocks for learning agents and specifically interactive multiagent adaptation of classification models as enumerated previously in Section 5.4. With these building blocks in place, their integration for the primary advisee role (cf. Section 6.3.1) and its complimentary advisor role (cf. Section 6.3.2) according to the specifications in Section 5.2 is discussed. Finally, Section 6.4 concludes the chapter with a summary of contributions.

## 6.1 A Role-based Architecture for Intelligent Agents

For the implementation of the knowledge management roles proposed in Section 5.2 that procure the foundation for a comprehensive learning element with knowledge management functions for an intelligent agent, the role metaphor has been adopted as an abstraction from and functional grouping of agent behaviours that are associated with a well-defined task. An abstract base class `Role` has been introduced which encapsulates a plan. It entails an appropriate behaviour script to effectuate the purpose of the role. This is shown shown in Figure 6.1. Each concrete role specifies the preconditions which need to hold for an activation of the role. Finally, each role exposes the means to register and deregister associated service descriptions with the directory facilitator of a JADE multiagent platform.

A dedicated agent class manages the conceivable roles that have been configured

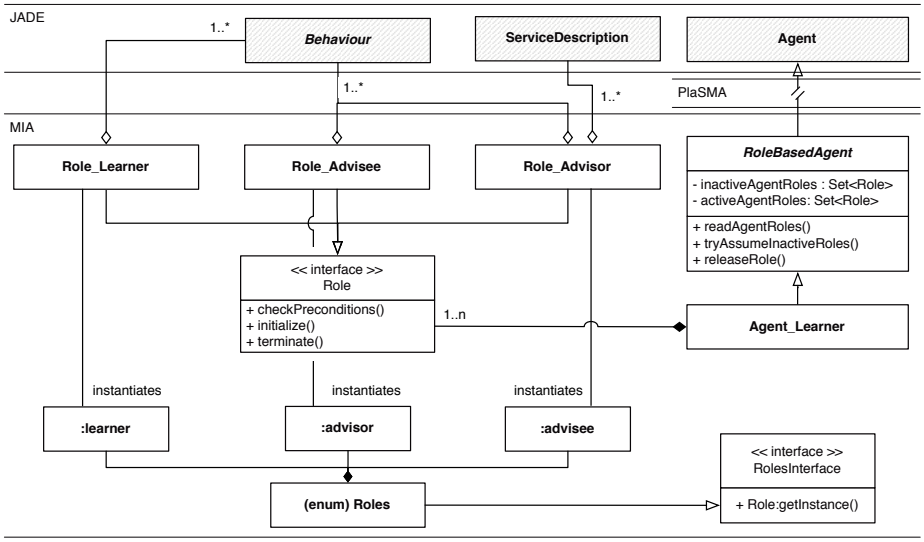



**Figure 6.1:** UML class diagram which illustrates the interplay of components in the implementation of the role-based architecture adopted by the agents participating in multiagent interactive adaptation.

for use by the agent. The roles of an agent can be either in dormant or active state. During agent initialisation, the complete role set of the agent is parsed from an XML-configuration file. Each encountered role is then instantiated and initially stored as dormant role. Once its initialisation phase is complete, the agent then tries and assumes any dormant role. Each role whose preconditions are met, is added to the active role pool and autonomously manages the scheduling of the appropriate agent behaviours to engage in role execution.

The agent class exposes a method which needs to be called in order to release a role and hence demote it back to the dormant state for future reuse. The respective method is called by the behaviours that implement a role in one of the following scenarios. First, a role may have succeeded in accomplishing its objective. In doing so, it may have fulfilled preconditions that are a prerequisite for downstream roles. Second, a role may have failed to accomplish an objective in spite of role-internal contingency handling mechanisms. In such a case, the role is rescheduled given that its preconditions are still met. Third, a role whose objective is the provision of a service such as learning advisory may have to stop its active operation when the basis for its service no longer holds.

The check for the activation of roles is not constrained to the agent setup and conclusion of other roles. Instead, it can be initiated by any active role while the direct role adoption is protected. Each agent needs to ensure that at least a single role remains active at all times to ensure that the agent remains responsive.

Each role, whether it refers to a knowledge management role as part of the agent learning element or a primary function in the agent's domain of application, serves an invariant objective or goal and by means of its associated behaviours codifies a stored plan to satisfy the objective. Therefore, the implemented reconsideration for role activation can be conceived as simple analogue to the deliberation step in the belief-desire-intention model (Rao & Georgeff 1997) wherein the agent

considers its options given its current intentions and beliefs about the state of the world and selects new intentions to adopt. Specifically, the set of active roles corresponds to the adopted intentions.

## 6.2 Implementation of the Learning Infrastructure

The preceding section has discussed a role-based architecture for the implementation of the necessary knowledge management roles in Section 6.3.1. Before delving deeper into the role implementations and hence the procedural aspects, it is necessary to first introduce the learning backend. It procures the essential building blocks for initial model acquisition and, more importantly, model adaptation, as discussed in Section 5.4.

When it comes to machine learning libraries suitable for academic use, a broad spectrum is available. A prominent general-purpose machine learning framework with a large community is the WEKA toolkit which is maintained by the the machine learning group at the university of Waikato (Hall et al. 2009). It provides a wide coverage of algorithms for classification, regression, clustering, and a comprehensive documentation (Witten et al. 2011). Another toolkit in this category focussed more towards visual programming of complex data mining applications is RapidMiner.

Tools such as Matlab and R also provide support for machine learning tasks. Other standalone toolkits such as the Structural Modelling, Inference and Learning Engine (SMILE) or the Fast Artificial Neural Network library (FANN) focus on learning algorithms for specific types of knowledge representations such as bayesian networks or artificial neural networks.

For the scope of this project, the Orange data mining and machine learning toolkit which has been developed at the Bioinformatics Laboratory at the University of Ljubljana, Slovenia has been adopted (Curk et al. 2005). It offers a functional range comparable to WEKA. The rationale for this choice has been twofold. First, due to the involvement of Možina et al. (2007) in the Orange development, Orange features extensions for argument-based machine learning. Specifically, it has an implementation of the ABCN2. Being written in Python, Orange is also easy to extend. WEKA classes were still used for data handling tasks and as outlined in Section 7.3, WEKA's database data importers needed to be extended to match the specific use case of multiagent interactive adaptation.

The UML class diagram (Booch et al. 2005) in Figure 6.2 illustrates the implemented core learning classes. The top part of this diagram is dedicated to interfaces which provide the functionality described in Section 5.4 as building blocks for model adaptation. To begin with, implementors of the `BaseLearner` interface need to provide functionality required for initial model acquisition. This includes accepting learning data from a data base. Besides credential settings, different table views for training data, test data, and a benchmark data set are specified. With access to data, the learner naturally exposes a method to trigger the invocation of the wrapped learning algorithm which then induces the classification model. This model is stored persistently by the learner class.

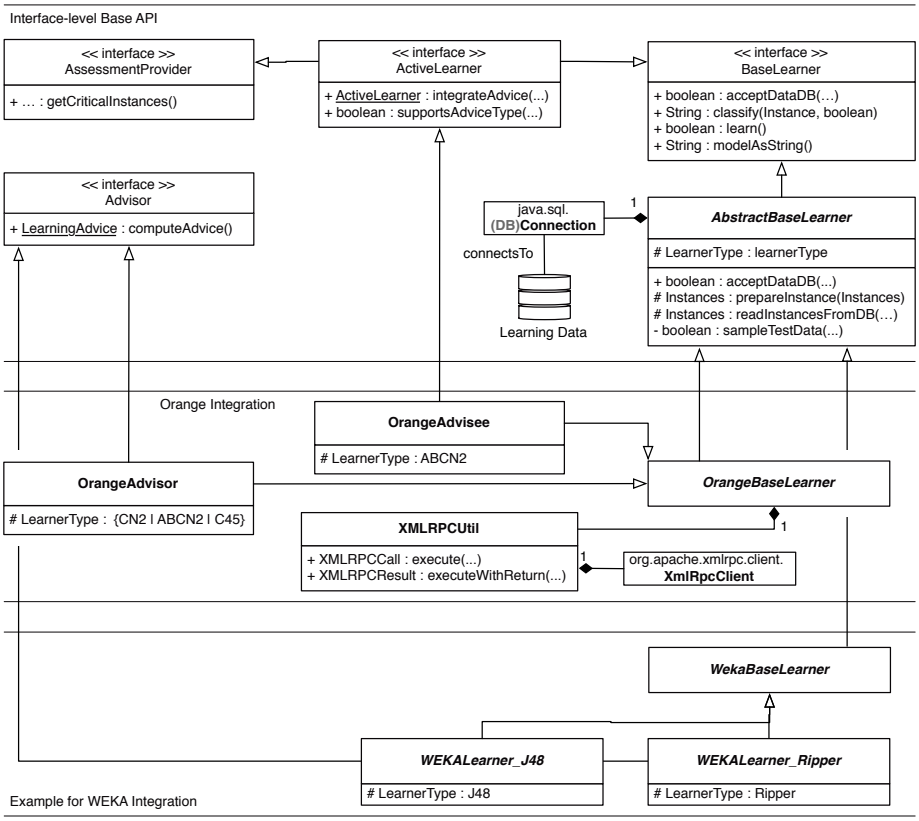



**Figure 6.2:** UML class diagram which illustrates the interplay of the most important components of the implemented extensible learner architecture for multiagent interactive adaptation on the Java side.

Hence, it is possible hereafter to demand the classification of new instances, a string representation of the trained model, or a basic performance assessment in terms of the general confusion matrix for either training, test, or independent benchmark data set. The classify function needs to support a strictly model-based classification. In the first step of the advice acquisition protocol from Section 5.4.2 (see also Figure 6.6), advisors are asked to classify the instance associated with a learning problem. This is not done to obtain a best guess at the most probable true class in which case a bet at the majority class is warranted in case of missing model support. By contrast, the advisee seeks to assess specifically whether the advisor model covers the learning problem at hand. Hence, this use case must be supported explicitly by classes implementing the base learner interface.

The `AbstractBaseLearner` class has been implemented to handle all interaction with the underlying data base which stores learning data. First, the class handles the split of the available learning data into training and test set according to a split parameter included in the agent configuration. The respective sampling is conducted on the database level using pl/pgSQL stored procedures. The functions for reading the data sets into main memory for further processing by derived learners are extensions of according WEKA procedures. These extensions were necessary as the implicit assumptions for data input from a database that were made by the WEKA developers did not match the scenario at hand. To be more

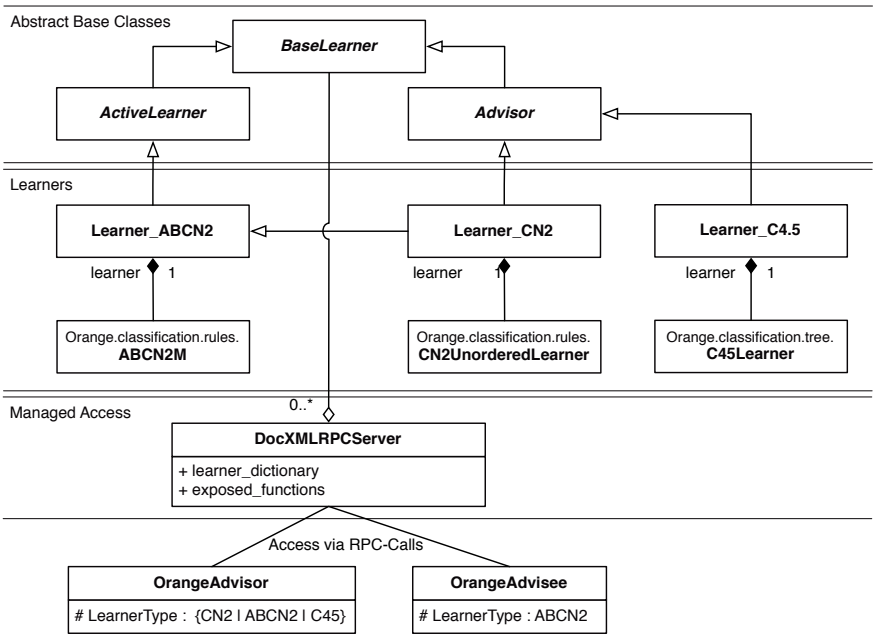



**Figure 6.3:** UML class diagram which illustrates the interplay of the most important components of the implemented extensible learner architecture for multiagent interactive adaptation on the Python side.

precise, WEKA assumes that all relevant domain information can be reconstructed on the fly while reading data. In a distributed learning scenario, it became clear that domain information needs to be provided explicitly in addition to the raw data. Otherwise, some learners may be unaware of specific feature values, as they do not appear in their learning data but may still occur in classification queries from peers or even their independent benchmark set. For this reason, a `DBInstanceQuery` class has been implemented which takes both a data and and an additional domain table as input. The latter explicitly states the possible values of nominal features which apply for the classification task at hand. In contrast to WEKA the imported domain is also invariant regardless of the order in which instances are read from the database. To conclude, the highlighted low-level efforts allow for unhindered learning, adaptation, and interaction among advisee and advisors.

Besides the `BaseLearner` interface, whose services enable initial model acquisition and classification for the purpose of an advisor adequacy assessment in advice acquisition, two additional interfaces have been specified that drive adaptation.

First, the `Advisor` interface exposes a method that procures `LearningAdvice` for a `LearningProblem`. While in the context of this research, said method is meant to encapsulate the computation of argument-based learning advice as introduced in Section 5.4.4, both input and output of the `computeAdvice()` method are specified in general terms by means of interfaces. Therefore, it is conceivable, for instance, for a future advisor to procure different types of learning advice and choose among them based on the current context.

Second, the `ActiveLearner` interface covers the utilisation of learning advice.

As the `integrateAdvice()` function returns an `ActiveLearner` instance, it allows for multiple scenarios of advice integration which have been discussed in Section 5.4.6. Namely, a direct modification of the existing model or, as pursued in this research, the training of a new model with the additional background knowledge received in the form of learning advice. Via the `AssessmentProvider` sub-interface an `ActiveLearner` also must be able to identify critical instances from its training data which can serve as basis for learning problems as described in Section 5.4.1.

The interfaces described thus far have been implemented based on the Orange library as shown in Figure 6.2. Technically, since Orange is a Python library, the Java interfaces presented before have been replicated in Python. As shown in Figure 6.3, concrete implementations for several learning algorithms have then been implemented such as the `Learner_ABCN2`, the `Learner_CN2` and the `Learner_C45`. Finally, a sound bridge between the Python and Java code bases was established by means of the XML-RPC remote procedure call protocol. A client-server architecture was established. It uses a Python server which manages the complete set of active learner instances for all agents that participate in multiagent interactive adaptation. The server exposes functions for learner instantiation and disposal. It also exposes the set of mandatory functions denoted in the interfaces in Figure 6.2. Once called in a remote procedure call, the respective function call is delegated to the correct learner instance belonging to the calling agent, and its results are returned to the caller. On the Java-side, the `Orange*` classes in Figure 6.2 act as wrappers for single remote learners. Each instance acts as a suitable XML-RPC client which is connected to the Python server specified in the agent configuration.

In addition to the aforementioned Orange learners, the UML class diagram in Figure 6.2 shows that two additional learners have been implemented based on the WEKA toolkit, namely the `Learner_Ripper` and the `Learner_J48`. From a technical perspective, this demonstrates the integration different machine learning backends into the adaptation framework. More importantly, it enables a greater degree of algorithmic heterogeneity with the inclusion of a tree-learner as complement to the rule learners.

The evaluation that is described in the next chapter will resort to a selection of learners from both learning backends.

#### Flexible Classifier Management Infrastructure

Section 2.3.1 discussed conceivable modes for the integration of learning, and by extension adaptation, and primary agent tasks. The reference implementation for interactive multiagent adaptation of classification models adheres to the principle of parallel learning and exploitation (cf. Figure 2.5 on page 21). In practice, a clean separation of productive and in progress classification must be introduced and be actively enforced by the architecture of the agent's learning element. As a consequence the `DSLearnerStore` has been introduced as a management service for classification models that internally acts as an abstraction from the shared datastore used by JADE-based agents. The management API thereby resembles in its operation a source code management system. Dedicated checkout methods are in place for the momentarily operationalised classifier, and, if present, the point of

origin classifier for an adaptation episode and the current state of the adaptation process. The same triad exists for commit methods. Also, purge methods have been implemented.

The design of the `DSLearnerStore` supports a potential multi-strategy approach to model adaptation as both commit and checkout methods for the non-operationalised model are qualified by adaptation strategy. Hence, the classifier storage infrastructure would also allow competitive parallel learning and operationalisation of the best result.

## 6.3 Implementation of the Knowledge Management Roles

Building on the role-based architecture for JADE agents and the learners from the preceding sections that implement the building blocks for classification model adaptation, this chapter now proceeds to provide details on the principal knowledge management roles proposed in Section 5.2.1 and Section 5.2.2. As it is the most complex role and the focal point of the proposed adaptation approach, the advisee role is presented first.

### 6.3.1 The Advisee Role

The external advisee role and its secondary internal advice integration role, both of which have been introduced in Section 5.2.1, have been implemented conjointly as a single agent role following the architectural pattern established in Section 6.1.

The advisee role has been realised by means of a nested finite state machine behaviour whose top-level structure is illustrated in Figure 6.4. The initialisation behaviour prepares the agent knowledge base for a new adaptation episode. Specifically, the classification model which is currently employed for productive use is retrieved from the model data store used by the agent and checked in as base model for the adaptation episode about to start (cf. Section 6.2 on page 140).

Once the preparations for the adaptation episode are completed, the advisee proceeds to the next behaviour which serves a dual purpose. When entered, the behaviour first acquires matching advisors for its classification task (cf. Section 5.3.2). This advisor acquisition is currently realised by means of a query which is directly posed to the conventional JADE directory facilitator agent which provides the yellow page services for the multiagent platform (Bellifemine et al. 2007, pp. 72). The design decision to enlist the platform directory facilitator service, rather than implementing a specialised yellow pages or advisory broker service as proposed in Section 5.2.3 has been made on pragmatic grounds. The service descriptions administrated by the directory facilitator were adequate to express advisory service details for the scope of the experiments conducted for the evaluation presented in the subsequent chapter. The direct queries which are placed to the directory facilitator can be replicated easily by means of a query interaction protocol (FIPA, Standard No. SC00027H, 2002) between the advisor and advisory broker which is executed in advance of the partner identification behaviour. Once the full set

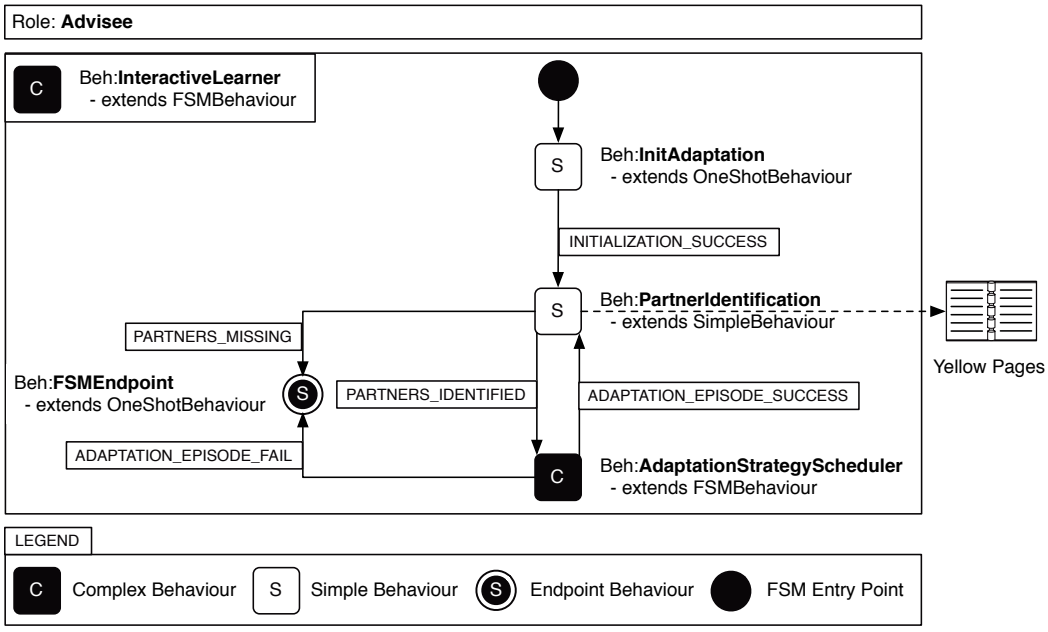



**Figure 6.4:** Composition of the top-level behaviour structure for the advisee role, implemented by means of a JADE finite state machine behaviour.

of potential advisors has been ascertained, the partner identification behaviour then proceeds to select the advisor panel for the upcoming adaptation episode. To that end, the behaviour is equipped with a partner selection strategy which is configured in the agent configuration.

Available strategies are implemented as functors that implement the `PartnerStrategy` interface with a single function which accepts the set of potential advisors as well as the set of last-recruited advisors (for the preceding adaptation episode) and returns the desired advisor panel. Two example partner strategies have been realised, one advocating the interaction with a single advisor per adaptation episode, the other modelling the opposite policy wherein all currently available advisors are added to the advisor panel. As argued in Section 5.3.2, the choice of the advisor panel shapes the adaptation process. The implementation therefore allows for a high degree of freedom of choice in strategy selection.

If the partner identification behaviour succeeds in assembling an advisor panel the agent engages in a new adaptation episode (cf. Section 5.3.1) by entering the adaptation strategy scheduler. Otherwise, the failure to assemble the advisor panel denotes the end of the momentary adaptation process such that the top-level finite state machine behaviour reaches its final state.

In this case, the top-level behaviour for the advisee role itself wraps up the adaptation process. It handles both the evaluation of the adaptation process in terms of achieved improvements in classification model performance and, as the case may be, handles the promotion of the final adapted model to the new production model. Only at this point in time, from the point of view of any active primary roles that rely on the classification model in question, said model is operationalised transparently using the `DSLearnerStore` such that the primary operation of the agent is not interfered with (cf. Section 6.2 on page 6.2).

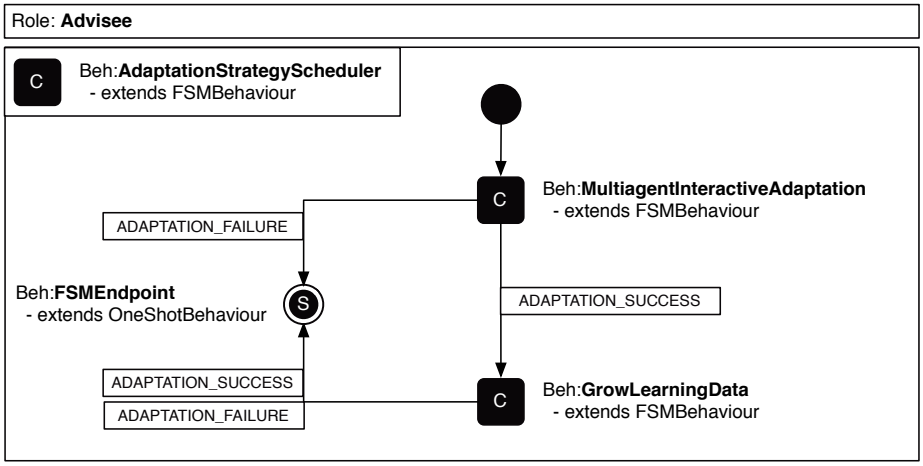



**Figure 6.5:** Composition of the adaptation strategy scheduler behaviour which constitutes the states entered in the adaptation top-level finite state machine behaviour shown in Figure 6.4 when an advisor panel for a new adaptation episode has been selected. The `GrowLearningData` behaviour constitutes a baseline strategy implemented for test purposes.

Returning to the case where a suitable advisory panel could be composed, Figure 6.4 shows that the advisee enters another nested finite state machine behaviour for adaptation strategy scheduling. Its inner composition is illustrated in Figure 6.5. The introduction of the scheduler behaviour as an intermediate layer between the the top-level finite state machine and the multiagent interactive adaptation as the strategy proposed in Chapter 5 of this thesis serves a dual purpose. From a pragmatic point of view, the possibility to schedule competing strategies is convenient with regard to an evaluation of the proposed adaptation approach against a baseline strategy. However, the architecture should be conceived as an opportunity for future implementation of a multi-strategy approach to the challenge of model adaptation, as called for by Michalski in Michalski (1993). Specifically, multiagent interactive adaptation as presented in this research could be complemented with other active learning approaches presented in Chapter 4. The implementation based on a finite state machine behaviour provides the flexibility to schedule adaptation strategies as called for by the respective use case.

### Implementation of the Online Search

The first adaptation behaviour which is called in Figure 6.5 comprises the actual implementation of the online search in the classification model state space which has been described in Section 5.3.1. Figure 6.6 illustrates the structure of the respective finite state machine behaviour. The diagram shows the interleaving of state exploration and advice acquisition which is characteristic for multiagent interactive adaptation as an interacting learning approach. When the advisee enters the adaptation control behaviour of the enclosing behaviour for the first time, a new adaptation episode and hence a new search process is initialised.

The control behaviour is thereby only designed as a wrapper for a concrete

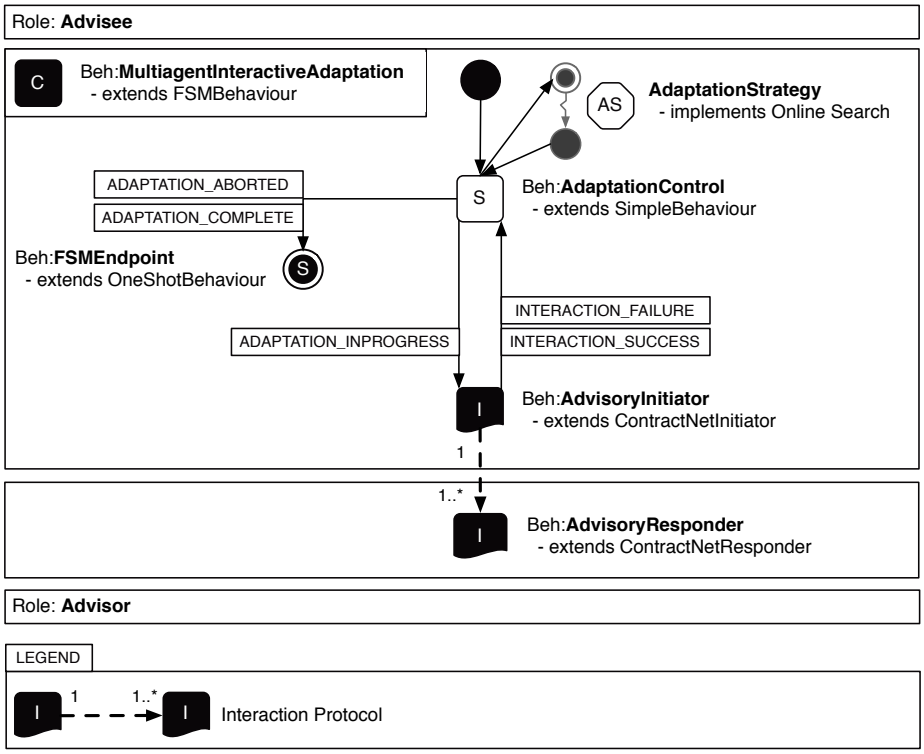



**Figure 6.6:** Composition of the interactive multiagent adaptation behaviour which manages the conduct of an adaptation episode, thereby relying on the embedded adaptation strategy component. The special notation of the reduced state machine end state connected to another re-entry state here and in the following Figure 6.7 marks the point where both state machine delegate control to integrate advice acquisition in the search process.

implementation of the `AdaptationStrategy` interface. For the evaluation in the following chapter, a hill-climbing search (Russell & Norvig 2010, p. 122) has been implemented that has been tailored to the search in model space. The concrete strategy to be used in interactive multiagent adaptation is again defined in the advisee configuration. On instantiation, a new search instance is presented with an appropriate model search problem. It comprises the original model providing the initial search state (cf. Section 5.3.1), a concrete implementation of the `ModelHeuristic` interface that can be used to check for reaching a goal state (`GoalTest` parent interface) and can assess the quality of a search state (`HeuristicFunction` parent interface, see also Section 5.3.1 on page 110), and finally a concrete implementation of the `LessonContentsStrategy` interface to be used to provide the learning problems to be addressed in state exploration.

The `HillClimbingStrategy` which has been implemented as a baseline of the model search procedure specified in detail in Section 5.3.1 is modelled internally as a finite state machine as well as shown in Figure 6.7. In fact, the finite state machine behaviour implementing multiagent interactive adaptation in Figure 6.6 and the strategy state machine have been interleaved via mutual delegation such that the advice acquisition from the advisor panel can be executed transparently from the point of view of the actual search implementation.

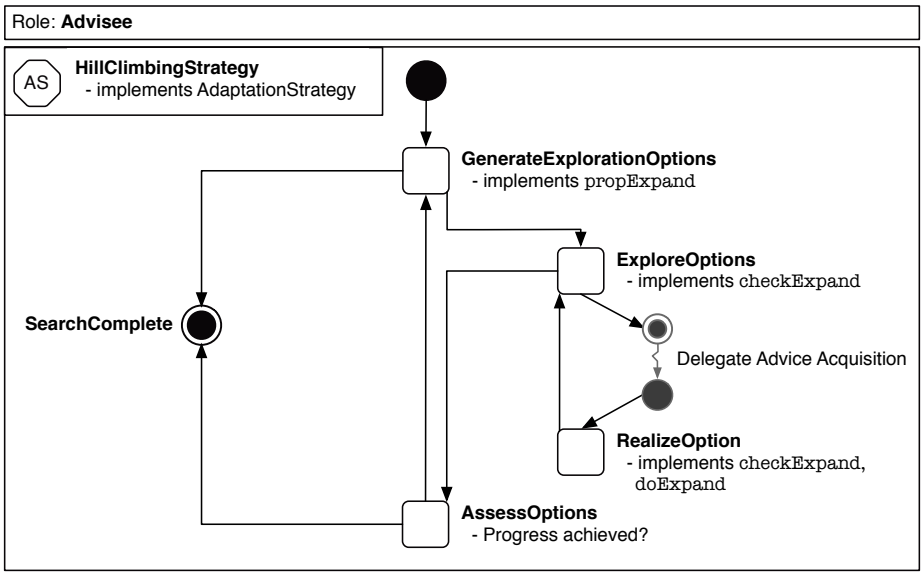



**Figure 6.7:** Illustration of the internal state machine implemented by the Hill Climbing search in order to reproduce the multi-tiered state expansion process which has been outlined in Section 5.3.1. This search procedure should be understood as complement to the behaviour illustrated in Figure 6.6.

An adaptation episode is started when the control behaviour first delegates execution to the hill-climbing strategy which in turn enters its initial state in which the exploration option for the state need to be ascertained with the help of the learner that has trained the current model. Hence, in this initial state, the first tier of state expansion which has been pinned formally in Equation (5.6) on page 106 is executed. When no learning options are proposed for state expansion, the search has been completed. Otherwise, the state machine enters a state which then manages the check whether learning problems that have been proposed can actually be handled due to availability of respective learning advice (see Equation (5.7) on page 106). Since said advice needs to be acquired from other agents which necessitates an interaction according to the protocol introduced in Section 5.4.2, control is delegated back from the search to the superordinate behaviour. Here the advisory initiator behaviour which is implemented in terms of a JADE Contract Net protocol (FIPA, Standard No. SC00030H, 2002) is entered. This protocol handles the interaction with the advisor panel and can be either successful (in which case at least one advisor actually procured learning advice) or a failure. Regardless of the result, the state machine behaviour returns to its adaptation control state which then immediately re-delegates control to the search procedure. Here, the expansion state (cf. Definition 5.8 on page 106) for the current search state is updated.

### Implementation of Argument Consolidation

In case of a successful advice acquisition, the specific type of received advice is matched against those types which can be accommodated by the learner used by the advisee. While multiagent interactive adaptation is built on argument-

based learning advice (cf. Section 5.4.3 and Section 5.4.4), it has been shown in Chapter 4 that other types of advice, for instance in the form of additional training instances as in active learning (cf. Section 4.2.1) could be incorporated as well. Acknowledging this fact, the handling of advice has been implemented with extensibility in mind. Hence, `LearningAdvice` is designed as an interface which is the implemented by concrete advice types such as `ArgumentAdvice`. Once compatible types of learning advice have been filtered, the remaining advice set is handed to an `AdviceConsolidator` instance. This component can be thought of as a complex functor which encapsulates the syntactic generalisation of argument-based learning advice which has been introduced in Section 5.4.5. This includes the conversion process from regular arguments as introduced in Section 5.4.3 into an extended form which includes the binarised argument representation from Section 5.4.5, the clustering process for similar arguments and finally the aggregation of arguments within a common cluster. The latter functions have been encapsulated in dedicated classes, namely an `ArgumentClusterer` and an `ArgumentReducer` respectively, such that the scope of operation of the AdviceConsolidator is achieved through functional composition. As with all critical aspects of the presented implementation of multiagent interactive adaptation, the argument consolidation is controlled with a parameter in the advisee configuration such that it is possible to opt-out of the consolidation process.

Once the processed set of advice for a particular learning problem has been computed, the associated potential successor model is finally induced based on the original training data, previous advice pool and acquired additional advice (see Equation 5.8). The circle constituted by the `ExploreOptions` state in Figure 6.7, advice acquisition in the superordinate finite state machine behaviour and finally the `RealizeOptions` state repeats until all learning problems have been addressed for the current state, advice has been acquired where possible and potential successor models have been induced. At this point, the search enters the `AssessOptions` state wherein it is determined whether the search has stalled or ought to proceed from a new state chosen from among the gathered potential successors. The assessment of model qualities is performed based on the objective function which has been specified for the search problem at hand in the form of a `ModelHeuristic` implementation as discussed before for the instantiation of the search problem. In case the best successor model improves upon the momentary model with regards to the chosen objective function, the successor is promoted. The search returns to the state `GenerateExplorationOptions` which heralds another process of state expansion. In the other case, the search process and hence the adaptation episode concludes.

As noted in the description of the top-level finite state machine behaviour for the advisee role, it is in its wrap-up code, called after reaching a terminal state, that the improvement that has been achieved over the course of a full run of multiagent interactive adaptation is assessed and a new production model is instituted as the case may be. Depending on the design choices with regard to the composition of the advisor panel and the dynamics of the learning environment, notably the addition of additional advisor over the course of adaptation, the resulting adapted model can be expected to be a result of a sequence of consecutive adaptation episodes, as laid out in Section 5.3.3 and specifically Definition 5.11.

### 6.3.2 The Advisor Role and Advice Composition

The implementation of the advisee role, as shown in the preceding section, features a complex behavioural pattern with several layers of nested finite state machine behaviours. This was due to the fact that an advisee needs to pursue a plan with multiple actively initiated consecutive steps in order to achieve the role's goal which is a significant improvement in model performance with regards to a custom objective function.

The advisee role, by contrast, is a service-oriented role. The preconditions for the adoption of this role are met when the respective agent has previously acquired a classification model whose performance is considered high enough to use it not only in primary, domain-specific roles. Hence, when the role preconditions are met and the advisee role is promoted to the active status, it initially handles the registration of its service with the entity that acts as advisory broker. In the current implementation, this is handled via a service registration request to the JADE directory facilitator.

As shown in Figure 6.6, requests to the advisee are subsequently handled by a single `AdvisoryResponder` behaviour that is implemented by means of a JADE `ContractNetResponder` behaviour. This behaviour models the responder side of the advice acquisition interaction protocol shown in Section 5.4.2.

The learning problems which are presented to the advisor are handled using the advisor's operative learner instance which supports the `Advisor` interface (cf. Figure 6.2). Hence, analogous to the case for the knowledge consumer side, the internal advice composition role introduced in Section 5.2.2 is filled by the embedded learner.

## 6.4 Chapter Summary

In this chapter, a comprehensive reference implementation of interactive multiagent adaptation as proposed in Chapter 5 has been presented. Due to its widespread adoption in the multiagent community, compliance with FIPA-standards for interoperability, flexibility in modelling agent behaviour, and its constituting the basis for the PLASMA simulation system, JADE has been chosen as multiagent platform for the implementation. A role-based architecture for the modelling of agent behaviour via functional decomposition into roles with specific objectives, preconditions, and associated service descriptions has been introduced. It enables the implementation of complex agents which are to assume different roles dynamically as the situation context demands. While generic in its approach, the architecture is particularly suited to realise the knowledge management roles which have been developed in Section 5.2.

The chapter also introduced the implementation of the learner components which encapsulate important building blocks for model adaptation as identified in Section 5.4. Specifically, the presented learning components implement the capabilities which have been described by the supplemental internal knowledge

management roles in the role system developed for multiagent interactive adaptation. Namely, the composition of learning advice on the advisor side and the subsequent integration of such advice on the advisee side. Based on the role architecture and procured learning components, the implementation of the advisee and advisor roles have been presented. For the advisor, the implementation has been realised with a special focus on modularity and future extensibility. In particular, the role design renders possible the addition of alternative adaptation strategies such as traditional active learning approaches, thus providing the foundation for a future context-sensitive multi-strategy adaptation approach.

Chapter 7

# Empirical Evaluation by Simulation Studies

The two preceding chapters introduced in detail the concept for an interactive multiagent adaptation of classification models and its reference implementation for Jade multiagent systems. Consequently, this chapter assesses the working capacity of the adaptation approach and carves out its characteristic traits, based on two use cases with real-world data sets.

Section 7.1 outlines the objectives for the evaluation. They have been derived from the research questions that motivated the presented research in Section 1.2.

Section 7.2 introduces the methodical approach for the evaluation which is based on extensive empirical simulation studies. The section addresses challenges in the design of the necessary simulation experiments with a large number of meshed system components, degrees of freedom in system parameterisation, and idiosyncrasies in the source data. In addition, design choices are motivated that have shaped the experimental setup. This involves the dispartment of an assessment of simulation studies from a macro perspective in terms of key performance indicators and a complementary detail investigation of the concrete classifier evaluation over the course of adaptation episodes, made possible by the white box characteristics of rule-based classifiers.

The choice of multiagent-based simulation as the means for the empirical evaluation, and specifically the use of the Plasma simulation environment, necessitated additional implementation efforts. Section 7.3 discusses enhancements to evolve the existing general-purpose simulation environment towards a flexible environment for the assessment of multiagent learning and, specifically, the process of interactive multiagent adaptation of individual classification models.

In Section 7.4, a first experiment series in this empirical evaluation is based upon the Fatality Analysis Reporting System (Fars) database. It has been provided by the US Department of transportation (National Highway Traffic Safety Administration 2016). Following a characterisation of the data set and applied preprocessing steps in Section 7.4.1, Section 7.4.2 presents a macro evaluation with seven different advisor panel configurations and sixty independent runs per

experiment, adding to a total of 420 simulation runs. Subsequently, Section 7.4.3 complements the assessment of the simulation study with a micro evaluation of paradigmatic experiment runs. The objective is a focus on the concrete change in an advisee classification model that is induced over the chronological sequence of an adaptation process.

Section 7.5 repeats the previous experimental setup on the forest covertype dataset whose instances describe 30 x 30 meter land cells obtained from US Forest Service (USFS) Region 2 Resource Information System data (Massive Online Analysis 2016). Both macro and micro-level evaluation findings are presented with cross-reference to earlier FARS findings.

The chapter concludes with a recapitulation of the core findings which have been brought up in the evaluation experiments.

## 7.1 Evaluation Objectives

The primary objective in focus during the empirical evaluation is the ascertainment of objective evidence for the fundamental validity of the proposed concept of interactive multiagent adaptation of classification models. This objective comprises several facets depending on the perspective view from which it is considered.

(1) From the perspective of agent-oriented knowledge management (Chapter 3), it needs to be substantiated that an environment can be formed in which an important category of knowledge processes, namely knowledge creation, is effectively furthered in a knowledge management environment populated exclusively by artificial actors.

(2) From the active machine learning perspective (Chapter 4), the design decision to adopt the technique of argument based machine learning and transfer it from an agent–human setting to an agent–(multi-)agent setting is subject to investigation as it has been employed as means to drive the advisee's internal learning processes from acquired learning advice.

(3) From a technology perspective, the capacity of the reference implementation of the presented concept (Chapter 6) is to be shown. The architecture should be shown to allow for flexible integration of induction techniques, thus allowing for algorithmic heterogeneity in investigated dynamic knowledge networks.

The evaluation study whose design is detailed in the following section has been devised to collect data to address these complementary perspectives.

## 7.2 Methodical Approach

Section 2.5 argued that due to the complex interactions among agents, it is often hard if not impossible to foresee and account for all conceivable runtime

interaction among these agents at design time (Jennings 2001, p. 38). As a consequence, the analytical prediction of those processes is often not practical. This holds especially true in scenarios where learning processes, and hence in particular interacting learning scenarios as explored with interactive multiagent adaptation of classification models, play a prominent role (Pawlaszczyk 2009).

### 7.2.1 Empirical Evaluation using Multi-agent-based Simulation

Lees et al. (2005) state that the use of simulation as an evaluation technique has traditionally played an important role in research and development in multiagent systems. Logan & Theodoropoulos (2001) note that "*the use of simulation allows a degree of control over experimental conditions and facilitates the replication of results in a way that is difficult or impossible with a prototype or fielded system, and it allows the agent designer or researcher to focus on a particular aspect of the system, deferring problems which are not central to the research*". The opportunity to focus specifically on key aspects of a system is also highlighted by Lees et al. (2005).

In the context of the presented body of work, the focus lies naturally on knowledge management processes. As these are conducted orthogonal to primary agent tasks the latter can be faded out for the purpose of a simulation study.

Section 2.5 argued that multiagent-based simulation presents an adequate and natural choice for a comprehensive micro simulation. Software agents are already the subject matter of agent-oriented knowledge management and the model structure can hence be preserved in simulation. In effect, the implementation presented in Sections 6.1 to 6.3 can be transferred with only minor changes into simulation and later into productive use.

This point holds in particular for the PlaSMA simulation system introduced in Section 2.5.3 due to its architectural objectives to facilitate adherence to the uniform agent design paradigm (Gehrke & Schuldt 2009).

Consequently, PlaSMA has been employed as basis for the design of a comprehensive experimental study, realised as a simulation scenario. The following section presents the concrete experiment design. It also gives an impression on the multitude of sometimes interconnected experiment parameters that entail both the environmental context in which the agents under study will be placed and their parameterisation with respect to knowledge management functions. The concrete parameterisation is to afford predictive simulation in order to assess the behaviour of emerging dynamic knowledge networks in which the proposed adaptation approach is exerted in controlled, reproducible conditions. In particular, the evaluation objectives stated in the preceding section need to be validated and concrete performance measurements based on a system of key performance indicators need to be assessed.

### 7.2.2 Experiment Design

The empirical study is comprised of two independent experiment series. In each, one of the data sets presented in Section 7.4.1 and Section 7.5.1 is employed as data reservoir. Each experiment is configured with a fixed random seed and is conducted 60 times. Each time, the random seed and the sampling of learning data for all participating agents change. A reference benchmark data set is kept invariant in size but also sampled for each experiment run.

Each experiment run hence creates prerequisites which model a specific moment in the execution of a multiagent system. With the respective system state as point of origin, the course of action for a single classifier adaptation episode is consequently modelled in the experiment run.

#### Knowledge Network Emergence

Each experiment in an experiment series is characterized by the number of participating agents and the adopted knowledge management roles out of the role system developed in Section 5.2 (cf. Figure 7.1 on page 154).

Seven agents participate in knowledge management activities. All of these initially assume the internal model acquisition role in order to train a classification model based on learning data obtained from a data warehouse source. Of these seven agents, a single one consequently assumes the advisee role (Section 5.2.1) while the remaining six agents assume the complementary advisor roles (Section 5.2.2). Thus, the knowledge network in the experiment is constituted.

#### Composition of Advisor Induction Algorithms

Each experiment series is composed of seven distinct configurations of the six participating advisor agents with respect to their employed induction algorithm. Hence, the algorithm itself is treated as a parameter of the experiment series.

Three different levels of algorithmic heterogeneity are configured with a spectrum ranging from

(1) three cases of complete homogeneity, i.e., six equally-typed rule and decision tree induction algorithms,

(2) three cases with a uniform distribution of two different algorithms, and finally

(3) a single case with a uniform distribution of all three employed base algorithms.

#### Invariant Advisee Parameterisation

First, the strategy for the composition of the advisor panel (cf. Section 5.3.2) is chosen such that all available advisors are immediately adopted as advisor panel for the first, and only, adaptation episode within the classification model adaptation process. The evaluation thus focusses specifically on the impact of a single adaptation episode and the online search process in the model space

encapsulated therein. This course of action is consistent with the evaluation of one specific situation in the life cycle of a multiagent system employing interactive multiagent adaptation.

Second, as a considerable group size of the advisor panel is effectuated, the advice consolidation as presented in Section 5.4.5 is enabled.

#### Invariant Advisee and Advisor Data Set Sizes

The size of the datasets initially procured for individual learning is configured with respect to the data reservoir used for an experiment. Concrete sizes are determined empirically in preliminary experiments for each task domain (see, for instance, Figure 7.4 on page 159). They are kept invariant throughout the 60 runs per experiment.

Advisors are provided with a significantly larger dataset than the advisee. This is a means to model the advisors as semi-experts in the application domain. This is based on the empirically funded observation that a more comprehensive body of data correlates with a high-grade classification model. The rationale for this modelling design decision is that it reflects a situation with 'experienced' advisors and a 'novice' advisee.

For each experiment repetition, the data is sampled according to configuration using unbiased random sampling for the advisee as well as the advisors.

## 7.3 Test Environment for Learning-centred Simulation Studies

Section 7.2 has identified predictive simulation experiments using multiagent-based simulation as a suitable means to assess characteristics and performance of interactive adaptation of individual classification models in multiagent systems within a controlled environment.

A PlaSMA simulation scenario has been set up to perform the empirical evaluation based on the implementation of the knowledge management roles discussed in Section 6.3. This scenario is sketched with its main components in Figure 7.1 on the following page.

Due to the specific focus on the learning element rather than the performance element of the agents under study, an abstraction from a traditional PlaSMA scenario had to be devised.

An important trait of the classification model adaptation in Chapter 5 is domain independence. Hence, instead of evaluating the approach only in the context of a specific application use-case, for instance in the context of knowledge management supporting autonomous logistics processes (Gehrke et al. 2010), the PlaSMA scenario needed to evolve into an experiment workbench, allowing for the performance of the proposed adaptation approach to be tested across a spectrum of tasks domains and classification use cases.

Consequently, the goal is to place simulation agents in a situation where the successful application of interactive multiagent adaptation of individual classification models can be observed under controlled and reproducible conditions.

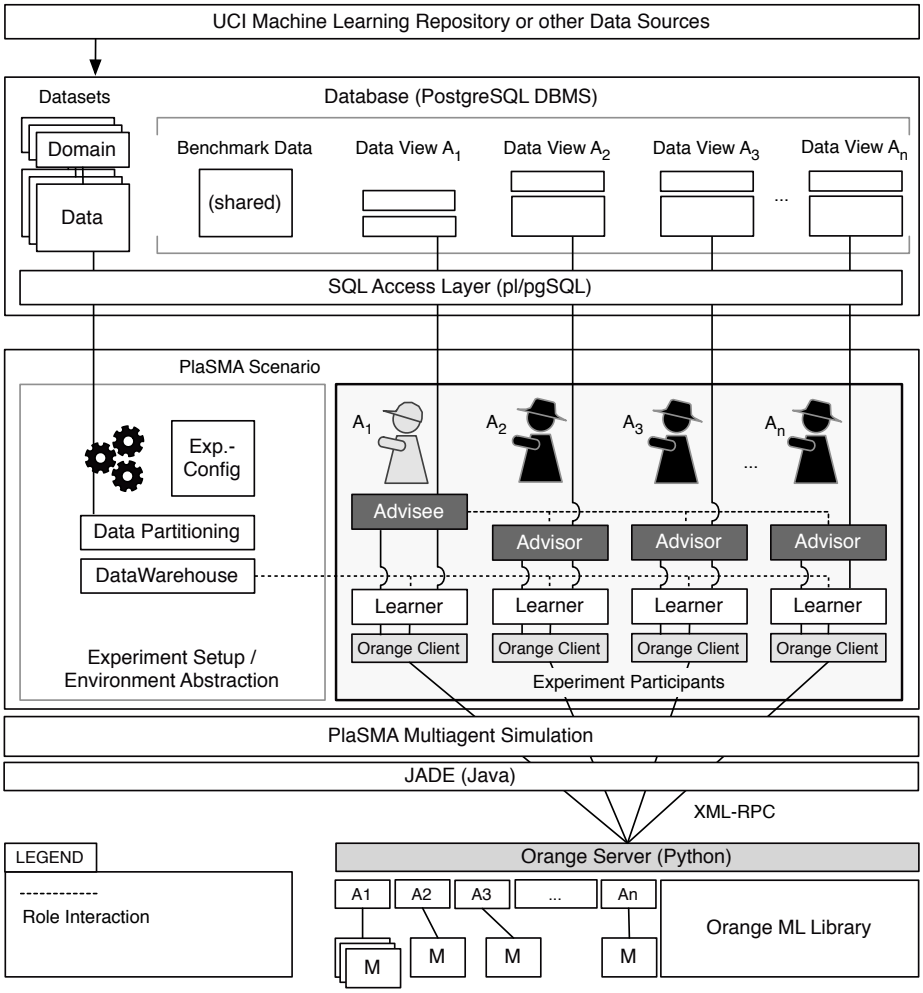



**Figure 7.1:** An conceptual overview of the components of the implemented experimental system for the evaluation of multiagent interactive adaptation.

From the point of view of agents participating in the adaptation experiment, the devised solution should be consistent with a real-world scenario. Several scenarios can be conceived as to the source of the data. Data could originate from the agent's own observation of its task environment. It could also originate from a dedicated data provider which can also be referred to as data warehouse. From the perspective of an agent learning element these options are equivalent. Thus, due to the focus of the empirical study on the characteristics and viability of interactive multiagent adaptation, the less complex data warehousing approach can be pursued.

### Plasma Data Repositories

Figure 7.1 illustrates that in response to the requirement to perform learning and adaptation experiments across different domains, PlaSMA has been extended significantly on the database layer with pl/pgSQL functions for the management of domain representations and data reservoirs from known domains. The domain

representation characterises nominal attributes in the feature vector and target classes. A range of sample domains has been adopted for an upload into the experiment database, sourced from the UCI machine learning repository (Frank & Asuncion 2010), the Massive Online Analysis (2016) repository, or first-hand sources such as the US Department of Transportation (National Highway Traffic Safety Administration 2016). Database views have been employed for feature vector filtering where adequate. Reporting functions have been implemented as well to obtain reservoir class distributions for the investigation of class imbalances.

The data sets to be uploaded to the experiment database are filtered through different preprocessing pipelines, depending on their origin and condition. Concrete measures that were necessary per domain are procured as part of the reports on conducted experiment series.

#### Implementation of a Data Warehousing Agent

For data warehousing, a dedicated agent has been implemented which is placed in the same PlaSMA simulation environment as the group of learning agents that participate in an experiment.

From the experimenter perspective, this agent is responsible for the sampling and partitioning of data for all learning agents (cf. Figure 7.1). To that end, the management agent has access to the collection of reservoirs of preprocessed data taken from different task domains outlined in the preceding paragraph. For a particular experiment, the data can be apportioned to the learning agents according to specification in the warehouse agent configuration. This encompasses both the total amount of instances which are to be placed at the disposal of a learner but also the class distribution of instances with respect to the target concept. Hence, various points of departure for adaptation experiments can be modelled.

From the perspective of the learning agents, the management agent acts as a data warehouse which provides them with views on their individual chunk of data. Specifically, in its warehouse role, the management agent procures mandatory database access credentials, and views for domain meta-data (shared), training data, test data, and benchmark data (shared).

The warehouse agent is implemented based on the same role-based architecture as the learning agents that incorporate knowledge management functions (cf. Section 6.1). The behavioural pattern for its data warehouse role is sketched in Figure 7.2. The initial sampling of data sets from the global data reservoir is conducted via pl/pgSQL stored procedures in the PostgreSQL database management system. Dedicated functions check whether all data requests for the experiment can be fulfilled. If so, a temporary table for the mapping from data records to learning agents is created. A reproducible sampling based on an unbiased Fisher-Yates shuffle (Durstenfeld 1964) is then applied to draw respective chunks of learning data. Finally custom database views are created for the learners.

Subsequently, a behaviour dedicated to the handling of data requests from clients is entered. As clients of the data warehouse, agents that have assumed the basic learner role for initial model acquisition find the warehouse service via a query to the directory facilitator and can then pose specific request for access to their

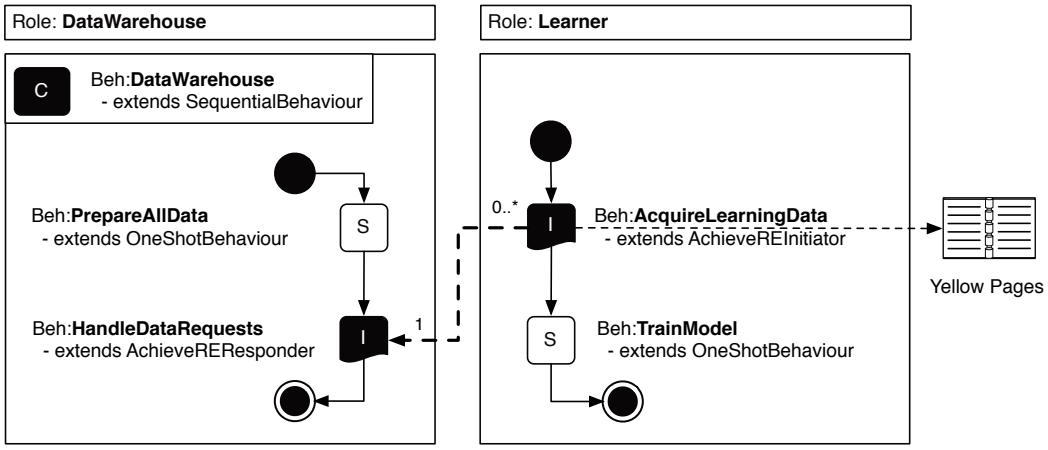



**Figure 7.2:** Behaviours for the data warehouse role filled by the management agent which also handles experiment setup as shown in Figure 7.1 and corresponding behaviour that constitute the learner role filled by all agents which participate in adaptation experiments as advisors or advisees.

learning data.

Using the `BaseLearner` capabilities of their respective embedded learners (cf. Section 6.2), both advisors and advisee can induce the initial classification model which serves as basis for advisory services or is to become subject to adaptation.

### Technical Configuration of the Experimentation Platform

The evaluation for this thesis has been performed on a single simulation machine with the following specification: 64bit Intel Core i7 CPU (4 Cores) with 2.8 GHz clock speed and 6GB main memory. The system runs Ubuntu Linux (16.04) and the Java Hotspot 64-bit Server virtual machine (version 1.8). A trunk SCM version of the PlaSMA simulation system has been used without functional changes to the simulation code base. The database management system which is used as a backend for both persistent storage of key performance indicators measured throughout experiment series and repository for learning data, PostgreSQL has been employed (version 9.1). The Python distribution which has been used for the execution of the Orange XML-RPC server was 2.7.3. The Orange machine learning library has been used in a development snapshot version, dating from 2011/10/19. This snapshot contains maintenance fixes to the ABML code base, addressing regressions in Orange 2.5 development cycle. These fixes have been kindly contributed by Martin Možina after personal communication. WEKA 3.6 has been used for data representation and as basis for reading data.

## 7.4 Empirical Study on the Fars 2011 Dataset

The first experiment series in this empirical evaluation is based upon data from the Fatality Analysis Reporting System (FARS). New compilations of FARS data sets are procured to the general public for assessment and research once a

| Dataset | Source | Features | | | Samples | Classes |
|---|---|---|---|---|---|---|
| | | ∪ | Nominal | Numeric | | |
| FARS 2011 Motorist | NHTSA | 43 | 41 (95.3%) | 2 (4.7%) | 60,762 | 4 |

**Table 7.1:** Overview of the FARS 2011 Motorist dataset used in the experiment series for this evaluation. NHTSA is an acronym for National Highway Traffic System Administration.

year by the US Department of Transportation (National Highway Traffic Safety Administration 2016). At the time of writing, the FARS records cover the period from 1975 to 2015.

For the experiment series at hand, the person data set for the year 2011 was chosen as input for careful conditioning to serve as domain and data reservoir for experiments. The dataset enumerates data for 72,495 persons that have been involved in recorded traffic accidents, specifically, individual-related data and accident circumstances associated with physical injury severity and applied treatments.

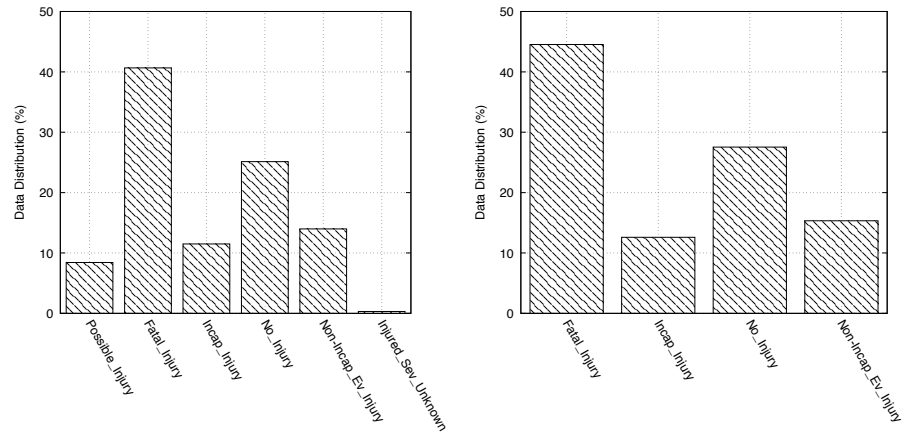
### 7.4.1 Data Set Characteristics and Processing

The raw data has been obtained in the DBF file format, a legacy format originally used with the dBase system. Using the Analytical Users Manual (National Highway Traffic Safety Administration 2012), feature names and possible values in the dataset have been transformed into a human-readable format to ease further processing. The summarisation batch processing features in the Open Refine (2016) toolset have then been employed for data cleansing.

In a further step, a part of the feature vector that characterized documented substance/drug use has been transformed from a format suitable for supposedly manual data acquisition by the authorities to a format that is reasonable for classifier induction. Specifically, the data from three unordered pairs of possible drug test entries, stating a test method and result, was turned into distinct nominal features for each test method and substance class findings (single or multi). As a consequence, the dimension of the feature vector for the FARS domain grew from the original 28 features to 45 features (see Table 7.1).

At this step in data preprocessing, it was acknowledged that the person data set can be partitioned in two groups of traffic stakeholders, namely motorists and non-motorists (National Highway Traffic Safety Administration 2012, p. 10). Said partition has two orthogonal dimensions. First, the records of the dataset can be partitioned, yielding 65,698 motorists and 6797 non-motorists. Also, the feature-vector can be compartmentalised for the respective group, i.e., 7 features covering amongst others seating position, or restraint issues, have been identified as motorist-exclusives, while a single on detailed specifically the non-motorist location in the recorded accident.

Both the feature transformation and partitioning of the original person data set have been implemented with a processing pipeline, built upon the Python data analysis library Pandas (2016). Of the two complimentary data sets that have been




**(a)** Classes (FARS 2011 Motorist, original) **(b)** Classes (FARS 2011 Motorist, reduced)

**Figure 7.3:** Histogram plots which break down the class distribution for the target concept for the dataset used in the experiment series for the evaluation.

produced, the much larger motorist part was chosen for further experimentation.

The processed FARS 2011 motorist data set has subsequently been subjected to closer scrutiny including experiment series test runs. These revealed a counter-intuitive conjunction of recorded racial affiliation and fatal injuries. The existence of a value for the race feature correlated almost perfectly with fatal injury. Hence, agents induced simplistic rules such $race = white \rightarrow fatal\_injury$. This circumstance could be explained by the supposed practice of data acquisition. Namely, that the death of a person at or as a delayed result of a traffic accident led to a detailed recording of person-related data which is not standard procedure for person with no apparent or only mild injuries. As a result of these observations, race was replaced with a synthetic feature that was modelled as having a weaker correlation with the injury severity target class.

The final step in data preprocessing involved a restriction to four out of the original six target classes (cf. Figure 7.3). As the class histogram on the lefthand side of Figure 7.3 shows, '$Injured\_Sev\_Unknown$' constitutes an extreme minority class (204 records amounting to 0.31 % of the motorist data set). This minimal permeation of the data reservoir was seen as problematic for the experiment series setup. The advisee would almost never encounter the instances in question and the advisees would be unlikely to develop a well-funded rule set including the minority class.

A second class '$Possible\_Injury$' was rejected for pragmatic reasons as learner tests showed that with the given feature vector even after preprocessing, the applied induction algorithms still had issues discriminating '$Possible\_Injury$' with other non-severe injury classes.

Consequently, the class histogram for the final FARS 2011 motorist data set is shown on the righthand side of Figure 7.3.

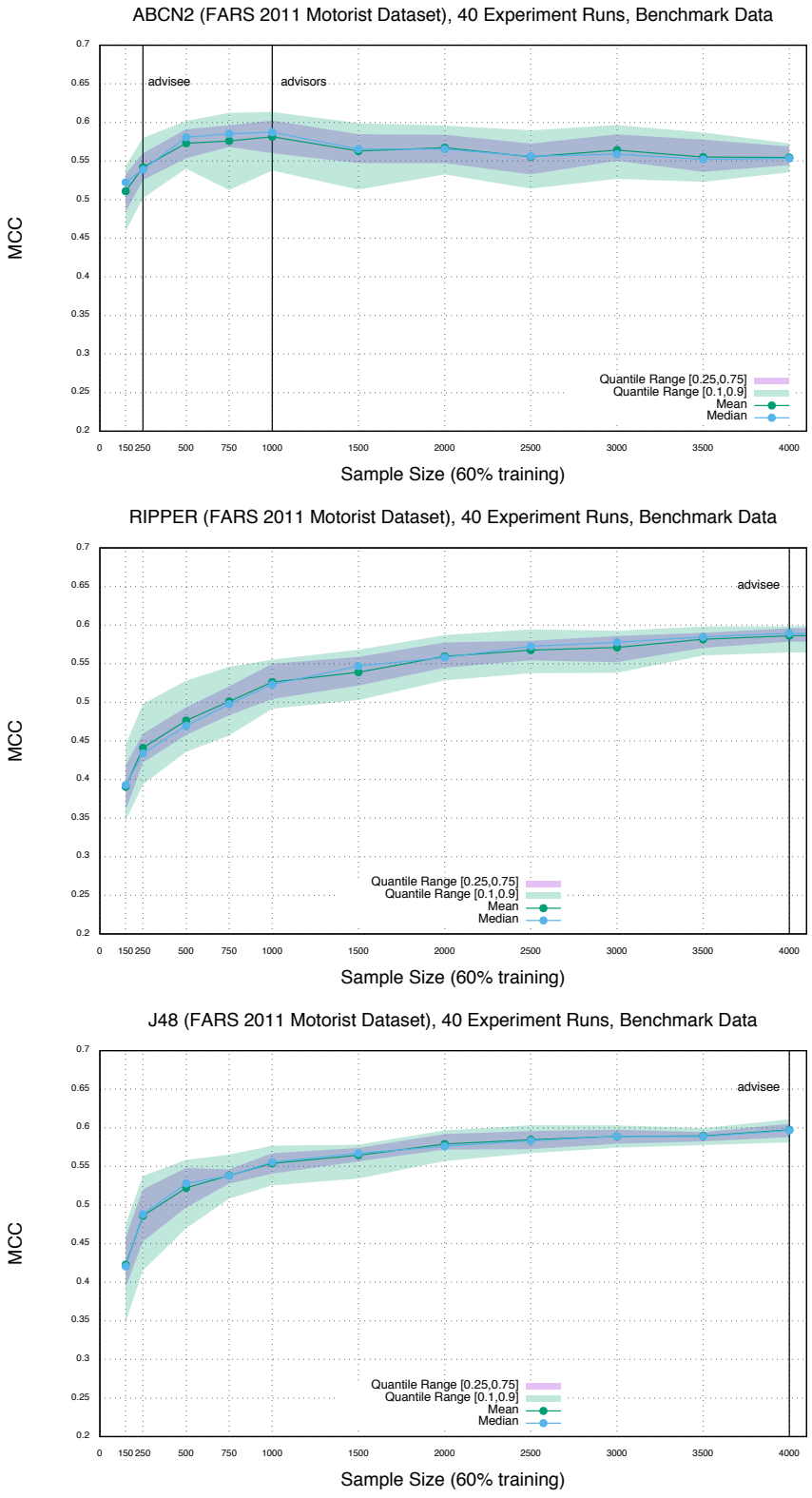



**Figure 7.4:** Learning curves based on 40 distinct experiments runs for each of the base classification algorithms that are employed by advisors (ABCN2, RIPPER, and J48) and advisees (ABCN2 only) in the experiment series for the FARS 2011 data set. The vertical lines denote the empirically determined number of instances to be procured to the respective agents in experiments.

#### Data Distribution among Advisee and Advisors

Once the repository of cleansed and processed data for the FARS 2011 motorist dataset was uploaded to the experiment database, a suitable assignment of instances for the actors involved in the planned experiment series had to be determined.

To that end, an empirical pre-study has been conducted which had learning agents using each of the classification schemes employed in the final study repeatedly with a growing sampling size of data randomly drawn from the global repository. In 40 repetitions for each data sampling size, the classifier performance on a very large reference data set of 20,000 samples was measured in terms of the Matthews Correlation Coefficient. The results of this pre-study are shown in the learning curves in Figure 7.4 on the preceding page.

Based on the learning curve plots in this figure, a reasonable size for advisor and advisee has been determined. Specifically, the advisor data size is determined by the point in the respective curve where growth essentially levels off. This indicates, that a further enlargement of the data set without further input augmentation efforts would demonstrably not yield better classifier performance.

For the advisee, the data set size had to be chosen such that an inexperienced learner is modelled. Consequently, as shown in the top-most plot in Figure 7.4, a point in the learning curve with a still steep inclination is chosen.

Beyond motivating data set sizing, the learning curves allow for the derivation of assumptions with respect of the effect of multiagent interactive adaptation of classification models in the given task domain.

First, the mean margin in classification performance between advisee and advisors is relatively small. As a consequence, model improvements are expected to be subtle. In fact, the approach under study is expected to yield better results where the curvature of the learning curves is less pronounced and the performance of the advisors better dominates that of the advisee.

### 7.4.2 Macro Evaluation of the Experiment Series

In the following, the analysis of both the performance and characteristic traits of interactive multiagent adaptation in an experiment series based on the FARS 2011 motorist dataset are shown from a high-level perspective.

It folds complex experiment runs, i.e., complete simulations, into single data points. These provide an impression on interactive multiagent adaptation in the particular situational context that has been constructed for a simulation run in terms of the accident data instances known to each agent initially. Although each such data distribution follows the same constraints defined by the experiment definition, seeded random processes ensure that across sixty consecutive simulation runs per experiment series and a given panel of advisor agents, a broad range of initial situations is encountered.

Figure 7.5 and Figure 7.6 cumulate a total of 420 distinct simulations spread across seven experiment series. The left side of the plot comprises 4 experiment

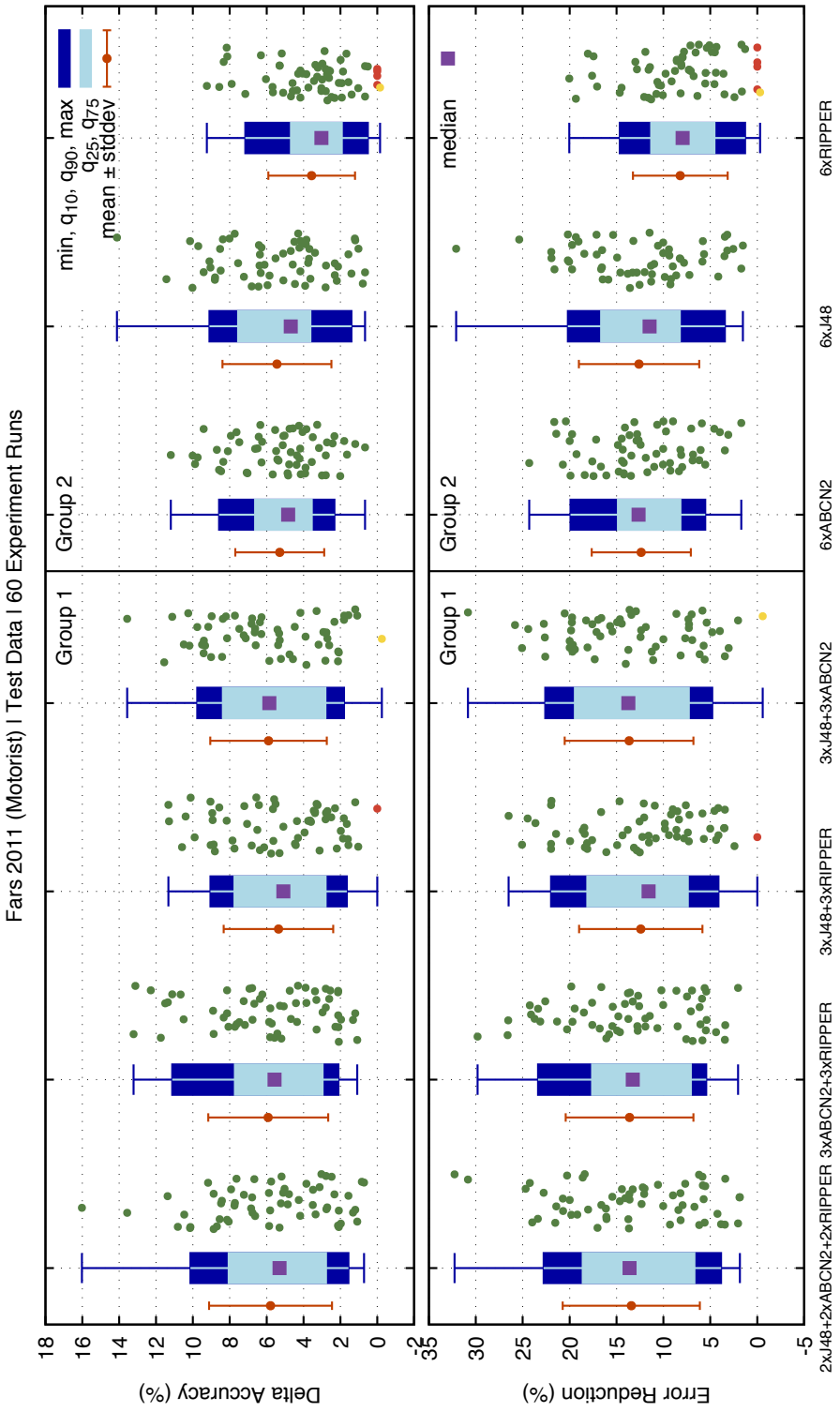



**Figure 7.5:** Comparative performance overview for multiagent interactive adaptation in an experiment series based on the FARS 2011 Motorist dataset. Each entry on the x-axis represents a distinct advisor composition in terms of the employed rule/decision tree learning algorithm with a fixed total of six advisors per experiment. Each experiment run, i.e., a simulation, contributes a single data point. The y-axis focuses on the MIA-induced effect on classification performance with respect to average accuracy.

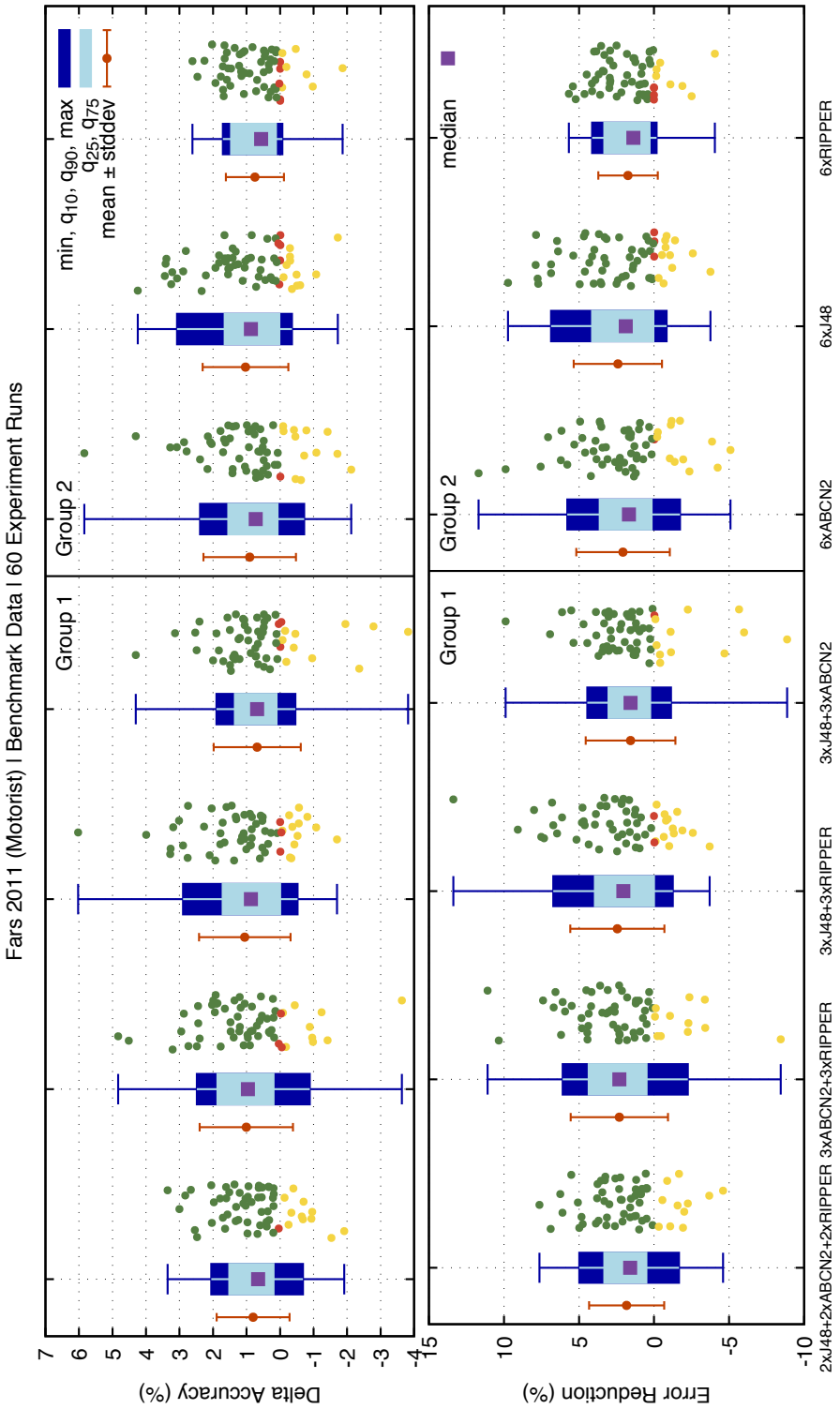



**Figure 7.6:** Results measured on the basis of a large-scale representative benchmark data set drawn independently from the complete universe of samples.

series with different induction algorithm mixtures of the six heads advisor panel. This is denoted on the x-axis of the plots. One mixture covers three induction algorithms while the other constitute equal-sized mixtures of a subset of two algorithms. Finally, the right side of the plot shows results for each algorithm in a homogeneous advisor panel.

### Observations on Experiments with Homogeneous Advisor Panels

With initial attention on those experiments that have been conducted with homogeneous advisor panels (i.e.,'Group 1' in Figures 7.5, 7.6 and complementing Table 7.2), performance measurements attest a discernible lead of the `6xABCN2` and `6xJ48` experiment series relative to the `6xRIPPER` series. For the advisee test data frame of reference, the ABCN2-advisors effectuate a mean delta in average classification accuracy of $\mu_{\Delta(acc)} = 5.29 \pm 2.41\,\%$ which amounts to an error reduction of $\mu_{err} = 12.37 \pm 5.29\,\%$. The median delta accuracy resides at 4.84 % with an inter-quantile range of $\langle\, q_{10}, q_{90} \,\rangle_{\Delta(acc)} = \langle\, 2.32, 8.59 \,\rangle\,\%$ (cf. Table 7.2). J48 results are comparable with a minimal edge in the mean delta in average classification accuracy but trailing in terms of standard deviation and median. That standard WEKA configuration of the RIPPER algorithm exhibits a weak performance in terms of the key performance indicators mentioned above. This is stressed visually in the overview plot in Figure 7.5. The distribution of data points for the homogeneous experiment series appear roughly normally distributed with a slight skew toward higher accuracies. While in both the `6xJ48` and `6xABCN2` series, outlier in the range of $\langle\, 10, 14 \,\rangle\,\%$ exist, the reputable scope of adaptation performance delta in terms of average accuracy lies between 0 and 10 % which corresponds to an error reduction between 0 and 20 %.

With respect to performance measurements based on the global benchmark dataset as reference, Figure 7.6 shows that the performance across all three experiments fails to stand up to the test results. Now, the `J48`-advisors exhibit the best results in what could be seen as a simulated operationalisation of the adapted advisee classifier. The data reads a mean delta in average classification accuracy of $\mu_{\Delta(acc)} = 1.03 \pm 1.28\,\%$ with an error reduction of $\mu_{err} = 2.41 \pm 2.95\,\%$. The median accuracy lies at 0.87 % with an inter-quantile range of $\langle\, q_{10}, q_{90} \,\rangle_{\Delta(acc)} = \langle\, -0.35, 3.07 \,\rangle\,\%$ or $\langle\, q_{25}, q_{75} \,\rangle_{\Delta(acc)} = \langle\, 0.02, 1.65 \,\rangle\,\%$. Hence, even for the most effective tried homogeneous experiment, only in about 75 % of the time, can an advisee that has completed an adaptation episode expect to see a positive effect for future classifier operation with an absolute delta in average accuracy of up to 3 % disregarding outliers.

**Finding**

It can be attested that for a large fraction of conducted runs across the three homogeneous experiment series, the investigated single adaptation episode per run is effective and induces the transition to modified classifier which, based on the benchmark dataset measurements, are expected to yield a performance improvement when operationalised by the respective advisee. For the FARS 2011 Motorist dataset, the observed classifier improvements are, however, evolutionary according to quantitative performance measurements. Consequently, initial proof for the effectiveness of the investigated adaptation methodology has been

| advisor panel | performance indicator | mean [%] | stddev [%] | min [%] | q25 [%] | median [%] | q75 [%] | max [%] |
|---|---|---|---|---|---|---|---|---|
| Data set: advisee-test, Experimental basis: 60 simulation runs | | | | | | | | |
| | *initial accuracy* | 57.297 | 4.813 | 46.993 | 53.726 | 57.741 | 61.301 | 65.476 |
| Group 1: heterogeneous composition of advisor induction algorithms | | | | | | | | |
| 2xJ48+2xABCN2+2xRIPPER | *final accuracy* | 63.087 | 4.84 | 48.08 | 59.89 | 63.538 | 65.987 | 72.248 |
| | *delta accuracy* | 5.79 | 3.337 | 0.714 | 2.78 | 5.301 | 8.056 | 16.028 |
| | *error reduction* | 13.426 | 7.306 | 1.858 | 6.696 | 13.613 | 18.603 | 32.259 |
| 3xABCN2+3xRIPPER | *final accuracy* | 63.215 | 4.297 | 48.08 | 61.79 | 63.744 | 65.83 | 72.248 |
| | *delta accuracy* | 5.917 | 3.253 | 1.087 | 2.965 | 5.577 | 7.725 | 13.223 |
| | *error reduction* | 13.61 | 6.802 | 2.051 | 7.085 | 13.27 | 17.614 | 29.817 |
| 3xJ48+3xRIPPER | *final accuracy* | 62.657 | 4.657 | 49.191 | 59.847 | 63.104 | 66.129 | 71.228 |
| | *delta accuracy* | 5.359 | 2.968 | 0 | 2.797 | 5.093 | 7.741 | 11.33 |
| | *error reduction* | 12.414 | 6.579 | 0 | 7.395 | 11.594 | 18.127 | 26.495 |
| 3xJ48+3xABCN2 | *final accuracy* | 63.197 | 4.654 | 48.08 | 60.176 | 64.078 | 65.987 | 72.068 |
| | *delta accuracy* | 5.9 | 3.161 | -0.256 | 2.817 | 5.855 | 8.376 | 13.566 |
| | *error reduction* | 13.665 | 6.868 | -0.584 | 7.293 | 13.732 | 19.43 | 30.832 |
| Group 2: homogeneous composition of advisor induction algorithms | | | | | | | | |
| 6xABCN2 | *final accuracy* | 62.587 | 4.757 | 48.635 | 59.46 | 62.418 | 66.409 | 70.759 |
| | *delta accuracy* | 5.289 | 2.413 | 0.658 | 3.548 | 4.837 | 6.631 | 11.2 |
| | *error reduction* | 12.366 | 5.285 | 1.7 | 8.19 | 12.642 | 14.851 | 24.3 |
| 6xJ48 | *final accuracy* | 62.738 | 4.618 | 49.191 | 59.751 | 63.004 | 66.404 | 70.119 |
| | *delta accuracy* | 5.44 | 2.954 | 0.658 | 3.63 | 4.688 | 7.554 | 14.12 |
| | *error reduction* | 12.599 | 6.406 | 1.525 | 8.249 | 11.484 | 16.634 | 32.09 |
| 6xRIPPER | *final accuracy* | 60.855 | 4.541 | 46.993 | 58.222 | 61.301 | 64.548 | 67.287 |
| | *delta accuracy* | 3.557 | 2.349 | -0.156 | 1.923 | 3.032 | 4.679 | 9.239 |
| | *error reduction* | 8.205 | 5.055 | -0.305 | 4.562 | 7.951 | 11.298 | 20.046 |

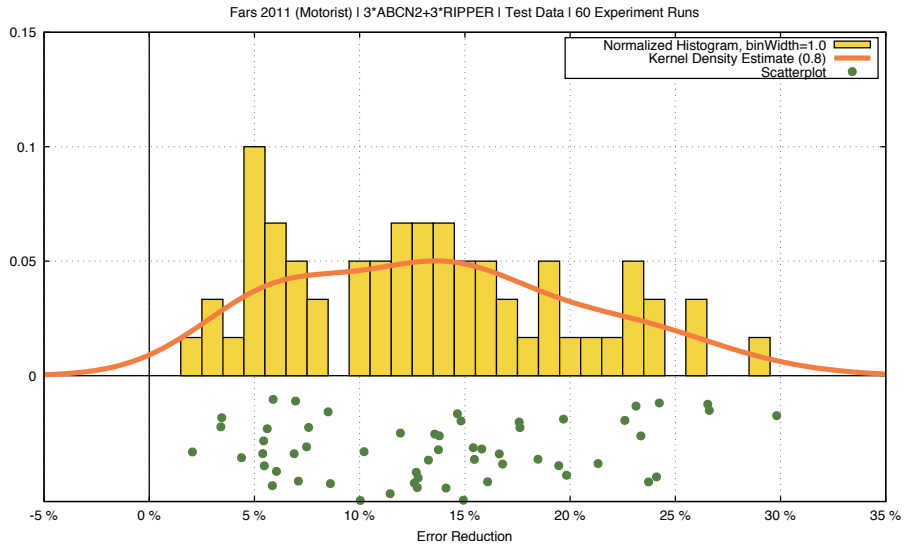
**(a)** Results measured based on the advisee test dataset.

| advisor panel | performance indicator | mean [%] | stddev [%] | min [%] | q25 [%] | median [%] | q75 [%] | max [%] |
|---|---|---|---|---|---|---|---|---|
| Data set: benchmark, Experimental basis: 60 simulation runs | | | | | | | | |
| | *initial accuracy* | 57.409 | 3.413 | 46.225 | 55.117 | 57.804 | 59.534 | 63.763 |
| Group 1: heterogeneous composition of advisor induction algorithms | | | | | | | | |
| 2xJ48+2xABCN2+2xRIPPER | *final accuracy* | 58.212 | 3.226 | 46.88 | 56.388 | 58.903 | 60.789 | 63.065 |
| | *delta accuracy* | 0.803 | 1.087 | -1.915 | 0.195 | 0.655 | 1.516 | 3.355 |
| | *error reduction* | 1.83 | 2.5 | -4.593 | 0.51 | 1.589 | 3.317 | 7.63 |
| 3xABCN2+3xRIPPER | *final accuracy* | 58.416 | 3.347 | 46.88 | 56.617 | 59.068 | 60.731 | 64.097 |
| | *delta accuracy* | 1.007 | 1.394 | -3.638 | 0.195 | 0.957 | 1.866 | 4.829 |
| | *error reduction* | 2.312 | 3.24 | -8.435 | 0.493 | 2.311 | 4.358 | 11.096 |
| 3xJ48+3xRIPPER | *final accuracy* | 58.462 | 3.485 | 47.093 | 57.012 | 59.453 | 60.668 | 64.364 |
| | *delta accuracy* | 1.053 | 1.364 | -1.701 | 0 | 0.869 | 1.714 | 6.026 |
| | *error reduction* | 2.449 | 3.135 | -3.705 | 0 | 2.05 | 3.946 | 13.377 |
| 3xJ48+3xABCN2 | *final accuracy* | 58.091 | 3.398 | 46.88 | 56.028 | 58.778 | 60.585 | 63.88 |
| | *delta accuracy* | 0.682 | 1.3 | -3.822 | 0.103 | 0.685 | 1.347 | 4.305 |
| | *error reduction* | 1.563 | 2.985 | -8.863 | 0.262 | 1.567 | 3.039 | 9.893 |
| Group 2: homogeneous composition of advisor induction algorithms | | | | | | | | |
| 6xABCN2 | *final accuracy* | 58.314 | 3.337 | 46.301 | 56.802 | 58.562 | 60.439 | 64.48 |
| | *delta accuracy* | 0.905 | 1.378 | -2.124 | 0.076 | 0.728 | 1.546 | 5.841 |
| | *error reduction* | 2.072 | 3.116 | -5.094 | 0.168 | 1.67 | 3.619 | 11.696 |
| 6xJ48 | *final accuracy* | 58.442 | 3.498 | 47.093 | 56.695 | 59.292 | 60.665 | 64.116 |
| | *delta accuracy* | 1.033 | 1.278 | -1.725 | 0.021 | 0.869 | 1.65 | 4.244 |
| | *error reduction* | 2.411 | 2.945 | -3.759 | 0.051 | 1.884 | 4.122 | 9.729 |
| 6xRIPPER | *final accuracy* | 58.158 | 3.365 | 46.225 | 56.585 | 58.449 | 60.322 | 64.454 |
| | *delta accuracy* | 0.749 | 0.868 | -1.865 | 0.124 | 0.567 | 1.464 | 2.612 |
| | *error reduction* | 1.738 | 1.981 | -4.048 | 0.297 | 1.378 | 3.318 | 5.674 |

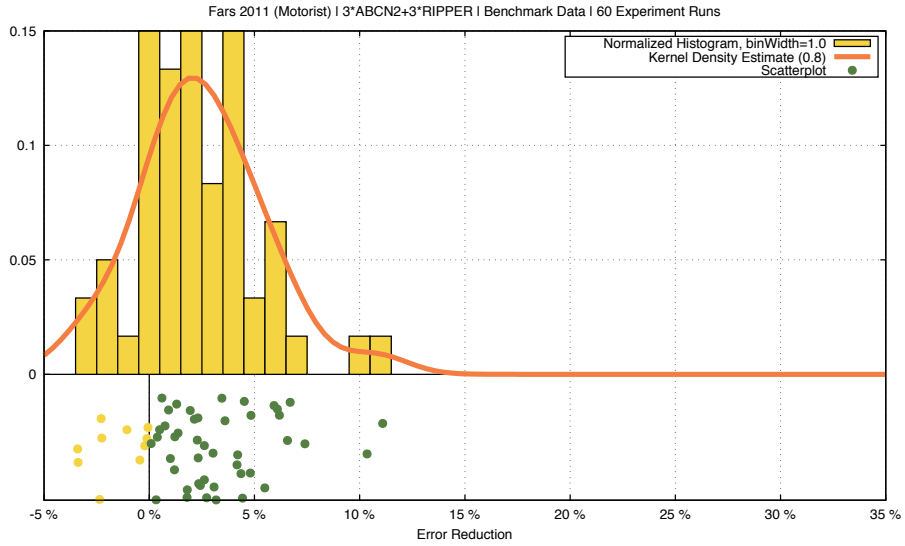
**(b)** Results measured based on a global benchmark dataset.

**Table 7.2:** Performance measurements for the FARS 2011 Motorist experiment series

presented but demands further differentiated analysis.




**(a)** Results measured on the basis of the respective advisee test data set limited in scope.




**(b)** Results measured on the basis of a large-scale representative benchmark data set drawn independently from the complete universe of samples.

**Figure 7.7:** Alternative representation of the error reduction measurements for one of the advisee configurations shown in the overview plots discussed earlier. The advisor configuration which has been chosen for closer inspection features two advisors using the J48 algorithm, two using RIPPER and another two using the ABCN2 algorithm. Each data point represents a distinct experiment run.

### Observations on Experiments with Heterogeneous Advisor Panels

Following the initial analysis for experiments with heterogeneous advisor panels, the focus is consequently put on the remaining heterogeneous settings (i.e., 'Group 2' in Figures 7.5, 7.6 and complementing Table 7.2). The respective results are thereby discussed relative to the earlier homogeneous results as baseline.

The performance measurements show a discernible lead of the `3xJ48+3xABCN2` and `3xABCN2+3xRIPPER` experiment series relative to the other experiment configurations. For the advisee test data frame of reference, the `3xABCN2+3xRIPPER`-advisors effectuate a mean delta in average classification accuracy of $\mu_{\Delta(acc)} = 5.92 \pm 3.25\,\%$ which amounts to an error reduction of $\mu_{err} = 13.16 \pm 6.8\,\%$. The median delta accuracy resides at $5.58\,\%$ with an inter-quantile range of $\langle\, q_{10}, q_{90} \,\rangle_{\Delta(acc)} = \langle\, 2.1, 11.12 \,\rangle\,\%$ (cf. Table 7.2). `3xJ48+3xABCN2` results are comparable with a minimal edge for the median delta in average classification accuracy but trailing in terms of standard deviation and means. The `3xJ48+3xRIPPER` experiment configuration exhibits a weak performance in terms of the key performance indicators mentioned above. This is stressed visually in the overview plot in Figure 7.5. Table 7.2 confirms that three out of the four investigated heterogeneous experiment setups improve on the homogeneous results presented earlier.

The result for the global benchmark dataset conform to previous observation in the homogeneous cases. An interesting observation though is, that the `3xJ48+3xRIPPER` configuration whose performance trailed the remaining heterogeneous configurations nonetheless exhibits the best performance retention when switching the frame of reference and hence considering the effect of a possible operationalisation of a modified classifier (cf. Table 7.2).

**Finding**

Due to the limited scope of the performance edge for the heterogeneous experiments and the experiment run sampling size, caution needs to be applied. Hence the wording that the conducted experiments suggest support for one of the assumptions that drove the development of interactive multiagent adaptation. The involvement of an as much as possible heterogeneous advisor panel of learning agents is a foundation for effective classification model adaptation.

### Discrepancy between Test and Benchmark Results

Thus far, the discussion addressed the effectiveness of the adaptation application as measured based on the comparatively small test set used by the advisee itself. To that end, Figure 7.6 presents the respective results based on a much larger benchmark data set. It can be used to understand the effect of using a modified induction model as operational model.

For the case of the Fars 2011 Motorist data set, it can be noted that the increase in average classification accuracy previously measured against test data fails to translate immediately to the benchmark data set (irrespective of the adaptive scale of the y-axis in the discussed plots). The overview plot suggests that the results on the test data ought to be characterized as over-confident. The spread of results across experiments now ranges from -4 % to as much as 6 % improvement ($\Delta$) of average classification accuracy including outliers of -1 % to 3 % when considering the $\langle\, q_5, q_{95} \,\rangle$ inter-quantile range. The range is both compressed and shifted below the 0 mark. Hence, the experiments show that a positive performance on the agent test set does not guarantee a benefit when the adapted

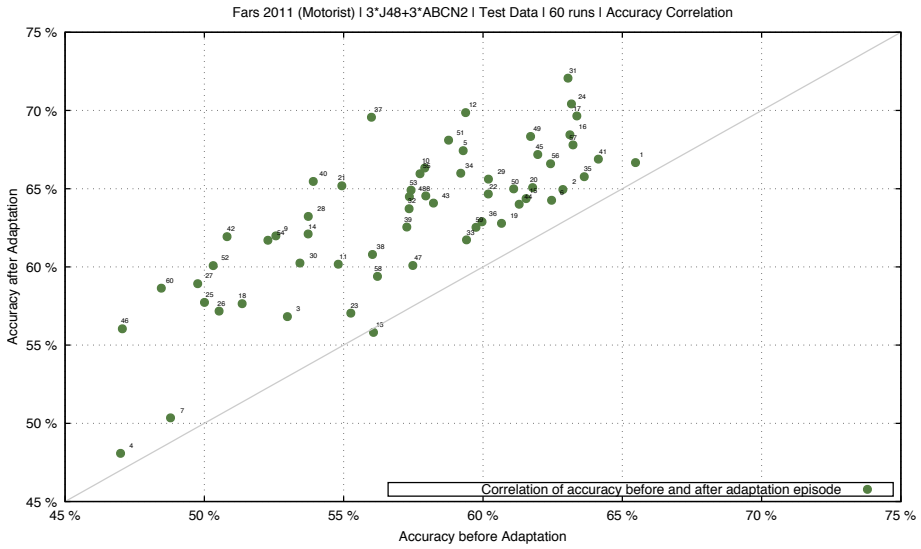



**Figure 7.8:** Correlation of classification accuracy on the advisee test set using the agent's original model (x-axis) and after model refinement (y-axis). The distinct data points have been labeled with their respective experiment run.

classifier is operationalised instead of the original version. While there is still a considerable chance of increased performance, the contrary is also possible.

A likely hypothesis to explain the situation is that the experiment design with respect to the use of the advisee test data set as evaluation reference to guide the adaptation process leaves room for improvement. Specifically, one can suspect that a somewhat more comprehensive and representative data set would mitigate the observed discrepancy between test and benchmark results.

In comparison, it is also interesting to accentuate that mixed advisor panels are affected differently when comparing test and benchmark performance. This effect is visualised best with the `3xJ48+3xRipper` advisor panel. While this configuration slightly trails the other heterogeneous configurations in Figure 7.12a, the configuration exhibits competitive performance on the benchmark data set, limiting negative adaptation effects with a comparatively high $q_{10}$ threshold while offering a competitive mean/media and high $q_{90}$ threshold.

**Finding**

There is empirical evidence supporting the assumption that the advantageous configuration heterogeneous advisor panel supports an induction model adaptation that translates comparatively well to unseen data, as exemplified in the conducted experiment series with the benchmark data set.

### Temporal Correlation of Average Classification Accuracy

Thus far, preceding aspects of the evaluation treated the application of interactive multiagent adaptation focussed on a bird's eye view on the conducted experiments and presented findings from the correlation of model compression and error

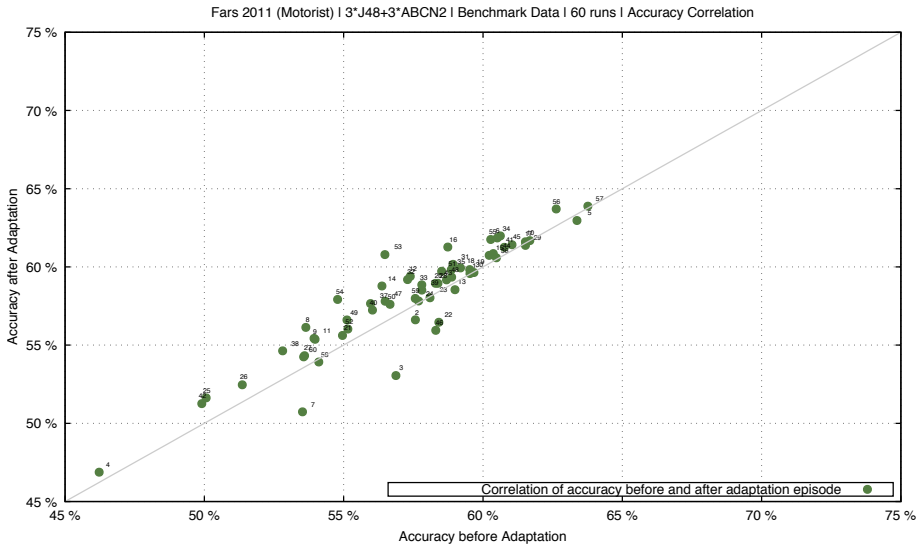



**Figure 7.9:** Correlation of classification accuracy on the global benchmark set using the advisee's original model (x-axis) and after MIA model refinement (y-axis).

reduction.

Now, the same experiments are evaluated from another angle. It is a first tap into the temporal dimension of each experiment run and hence seeks to elucidate the model refinement effects over the course of a complete adaptation episode with its discrete steps (cf. Section 5.3 on page 102).

Figure 7.8 presents a correlation of the average classification accuracy of the original advisee agent model when engaging into a model refinement episode and the adapted version operationalised thereafter. As the adaptation framework guarantees that its application will yield a refinement result that is no worse than the input, all data points in the upper plot reside on or above the diagonal.

The coarse shape formed by the data points in the plot does on a limited scale empirically confirm expectations as to the effectiveness of adaptation application for different initial scenarios in terms of the performance of the advisee's original model. Up to a threshold of about 55 % original accuracy, the model adaptation process almost completely succeeds to raise accuracy onto the $\langle\, 55, 65 \,\rangle$ % percent range with the maximum resulting accuracy seemingly linearly correlated to the original value. Beyond the aforementioned 55 % threshold, the growth in accuracy eventually levels of at around 70 % average accuracy. Specifically, it should be noted that with the agent test set as reference, the investigated setup succeeds in raising the classification accuracy to a level that falls in the broad range of between 55 % and 70 % irrespective of the initial situation.

Another expectation that is confirmed on a limited scale in the plot in Figure 7.8 is that the better the performance of the original model, the smaller the refinement gains. This is evident in the upper right quadrant of the plot where the data point adapt ever more tightly to the diagonal.

A comparison of the plots in Figures 7.9 and 7.8 shows that the overall characteristics of the refinement on the FARS 2011 Motorist dataset are quite different

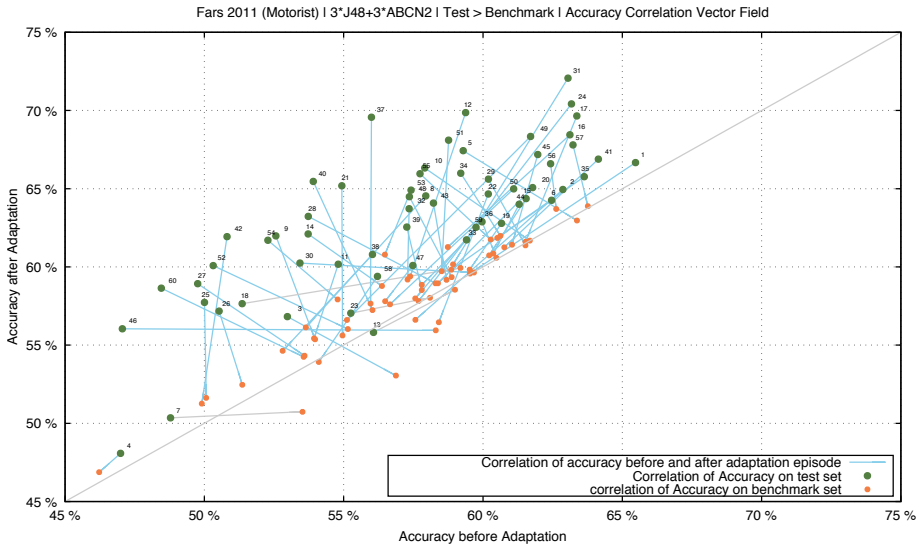



**Figure 7.10:** A complex correlation plot which presents three interrelated analyses within a single graph. First, it correlates the advisee classification accuracy before and after a round of multi-agent interactive adaptation for sixty independent experiment runs as measured based on the respective advisee test data set (green data points). Second, the same correlation has been applied with the larger benchmark data set as basis of measurement (red data points). Finally, data points which belong to the same experiment run have been connected with colour-coded lines. A grey line indicates that the test dataset led to an underestimation of true classification performance, a blue line indicates the opposite case.

when measured based on the global benchmark data set. The data point cloud is shrinked on both axes. This suggests that the comparatively small size of the advisee test set is often not a good sample draw in order to approximate the characteristics of the agent environment. Paradigmatic examples for this issue are four data points in Figure 7.8, corresponding to the experiment runs #46, #18, #1, and #24.

The additional plot in Figure 7.10 offers a visual representation of this situation by superimposing the data points for both the experiment run-specific agent test sets and the considerably larger benchmark data set and connecting both data points for the same experiment run.

As a result, it is shown that in experiment runs such as #46, #28, #18, but also #30, or #5, the measurement for the advisee test set significantly underestimated the actual accuracy of the original advisee model by up to 10 %.

At the other end of the spectrum, experiment runs such as #1, #41 or #35 exhibit the inverse problem with an overestimation of actual accuracy. Without pointing out specific further instances the plot also reveals that the same characteristic error pattern also applies for the estimation of classification accuracy after model refinement.

This observation is significant as it highlights potential for optimisation of the adaptation framework. Specifically, as the delta in classification performance drives both the path selection and termination decision in the local search

that constitutes the basis of the top-level control loop (cf. Chapter 5.3), highly fluctuating estimations inhibit the search for acceptable local optima in the search landscape.

### Empirical Analysis of Weak Test/Benchmark Performance Alignment

In order to empirically explain the discrepancies in performance measurements and subsequently derive advice for future improvements, a selective deviation of the experiment configuration from Section 7.2.2 has been investigated paradigmatically for the `3xABCN2+3xRIPPER` advisor panel. This additional experiment series differs from the original variant only in the scope of the advisee test set.

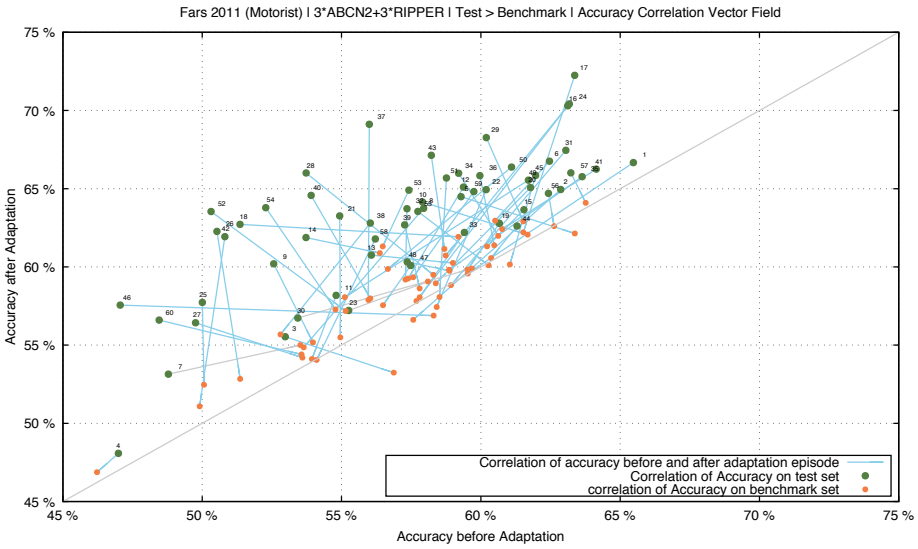
The rationale for this design is a confirmation of the assumption, that the small size of the advisee test set with its 100 instances (i.e., 0.4 % of the total advisee data set), although large in proportion to the training set, is still too small for a sensible approximation of the much larger benchmark data set (20,000 instances).

For the experimental investigation of the above assumption, the advisee test set has been boosted to 850 instances (i.e., 0.15 % of the total data set). In isolation, this weighting between training and test data is not rational from the point of view of the advisee. Nevertheless, with the given experimental context, this configuration is a pragmatic way to emulate the following scenario: Instead of limiting itself to self-sufficient approach based on its own limited test data set, the advisee is able to employ a larger, community-based benchmark data set assembled from contributions of all agents partaking in the dynamic knowledge networks in which individual classification model adaptation takes place.

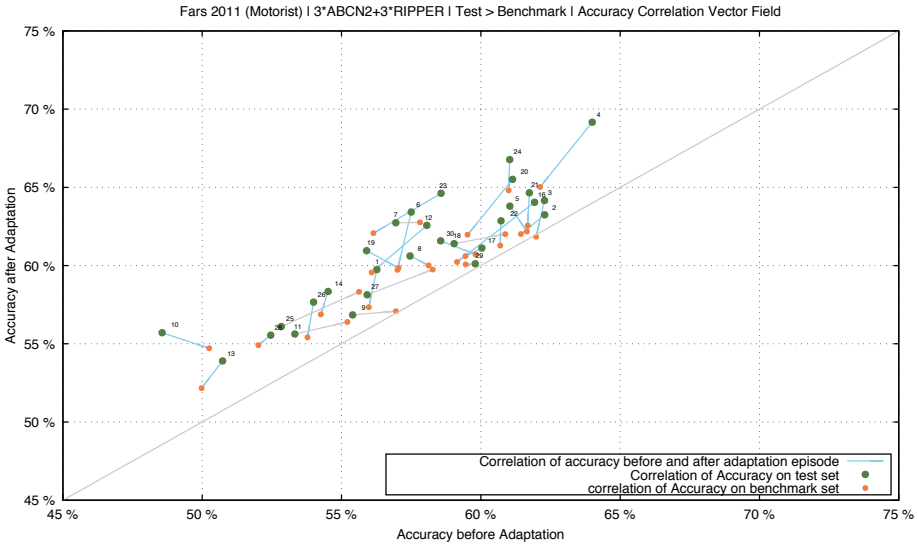
Figure 7.11 shows two plots which directly oppose the correlations for average classification accuracy before and after application of an adaptation episode for the original experiment series on `3xABCN2+3xRIPPER` and the community reference test dataset version (lower plot). These plots immediately confirm the assumption posed initially in this section. The mean deviation of test and independent global benchmark results are reduced to a significant degree. This means, that the measurements on the test set are more faithful estimators for performance once an adapted classification model is operationalised by the advisee. In particular, the detrimental effect also discussed earlier in this section, that operationalisation of a new model might even have a negative impact (visualised by data points below the diagonal in Figure 7.11), can be almost eliminated.

**Finding**

The results discussed above offer a clear hint on enhancing the baseline robustness of the investigated adaptation approach. At the same time, the findings from this additional empirical study confirms the system of knowledge management roles which has been developed in Section 5.2. This holds specifically for infrastructural roles within dynamic knowledge networks such as the knowledge benchmarking role introduced in Section 5.2.4. Indeed, the experiment series variation discussed above serves as an emulation of the benchmarking role and motivates its elaboration as part of future work (cf. Section 8.2).




**(a)** Results for the original experiment configuration.



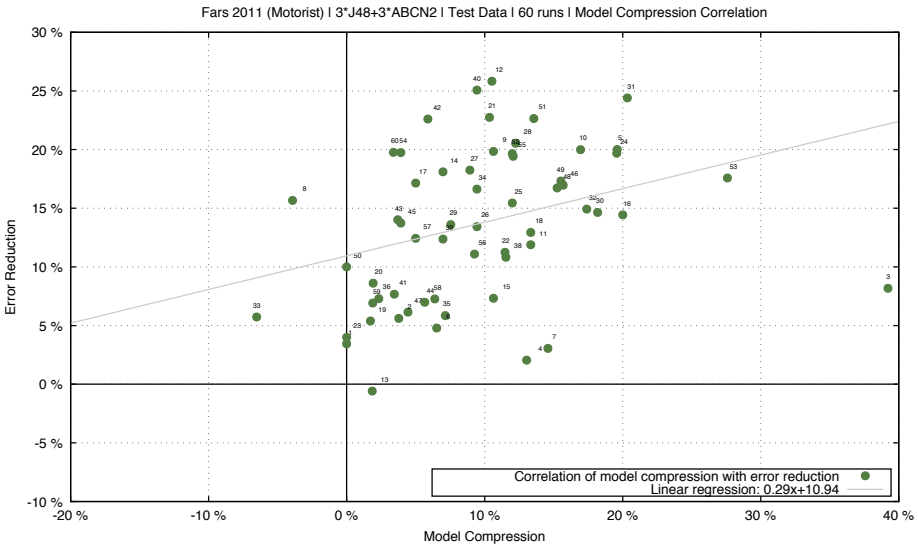

**(b)** Results for a modified experiment configuration with a five times larger advisee test set.

**Figure 7.11:** A complex correlation plot which presents three interrelated analyses within a single graph.

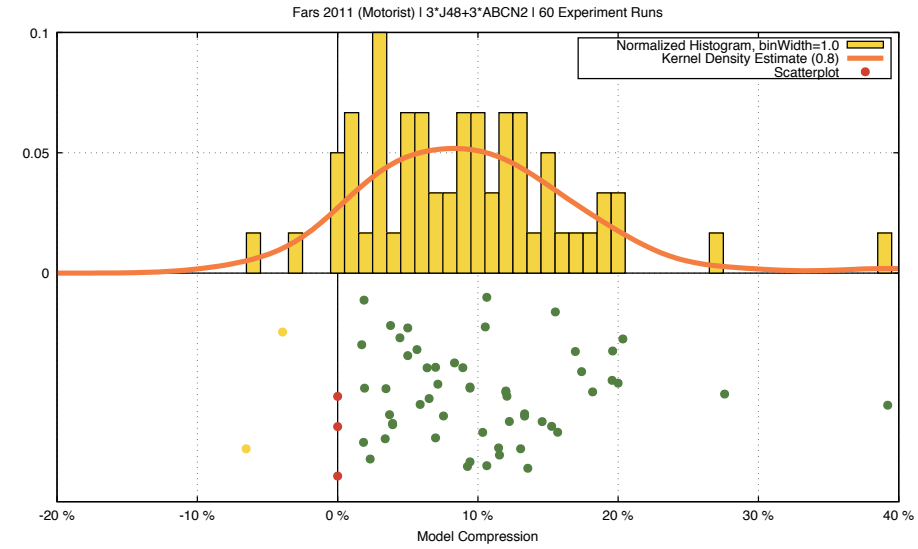
### Correlation of Performance Improvements and Rule Set Compression

The following section takes into account the composition of the classifier model which is refined by the advisee agent within each experiment run. As the advisee employs ABCN2 as rule induction algorithm, the refinement operates on a discrete set of rules. For such a set, its volume comprises an interesting key performance indicator for model complexity and, more importantly, indirectly on the generalisation potential of the refined model.

The line of thought leading to the latter statement is as follows. The less rules




**(a)** Results measured on the basis of the respective advisee test data set limited in scope. The latter explains the larger deviations from the linear regression function ($f(x) = 0.29 \cdot x + 10.94$) as compared to those for the benchmark set below.




**(b)** Histogram of percentaged rule set compression for a complete experiment series.

**Figure 7.12:** Correlation of the deltas in classification accuracy and model complexity in terms of the number of contributing rules that have been effectuated by multi-agent interactive adaptation. For the Fars 2011 Motorist data set that forms the basis for these experiments, the plot confirms a linear relation among these two key performance indicators. This conclusion is also reflected by both the linear regression and the curve of the weighted cubic splines approximation.

constitute a rule induction model to perform classification on a given level of accuracy, the more generality can be assumed for the remaining rules. At the same time, the probability of such rules being indeed widely applicable and not biased to suit only the input data used for learning, increases. Hence, the model can be theorised to be less likely to be over-trained.

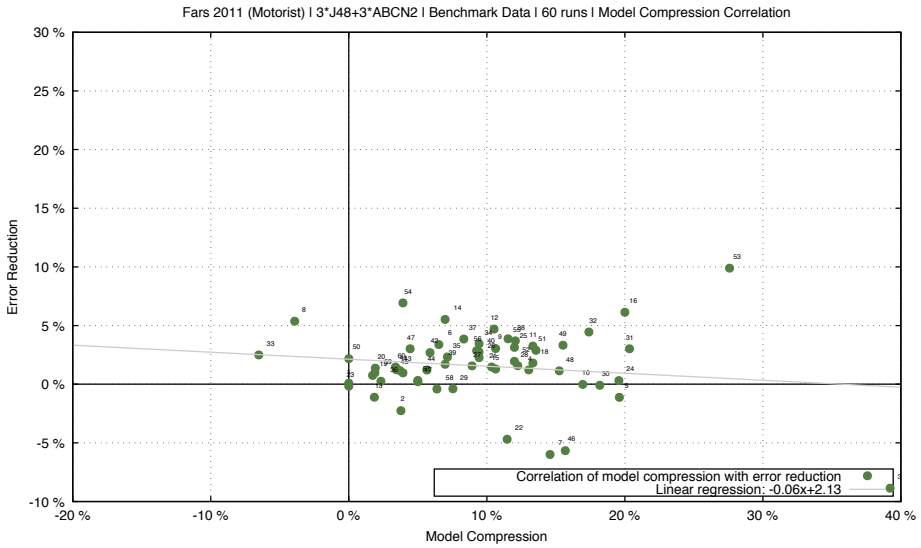



**Figure 7.13:** Result measured on the basis of a large-scale representative benchmark data set drawn independently from the complete universe of samples. Notice that the linear regression function $f(x) = -0.06 \cdot x + 2.13$ sheds a more conservative estimate of the MIA effect in real-world situations.

The plots in Figure 7.12 consequently correlate the error reduction key performance indicator with a new key performance indicator termed model compression. It measures the percentaged reduction in rule set size induced by model adaptation. While Figure 7.12a covers the situation with respect to the respective advisee test set, the lower plot in Figure 7.12b shows a histogram of model compressions.

**Finding**

A glance at Figure 7.12a highlights that there is a positive correlation pattern. This is expressed in the inclination of the linear regression function. However, it stands out that the average distance between data points and the regression line is large on both sides of the function. Hence, the experiments yield empirical support for the expected correlation but also caution not to overstate the results.

This is true in considering the benchmark result in Figure 7.13. The restraining effect of a change of the measurement reference system that has already been observed in Figure 7.29 on page 199 is reflected again. While the data points in the correlation plot are drawn substantially closer to the regression line, the inclination nearly turns flat.

Before taking a closer look at the correlation results themselves, the model compression on the x-axis of the plots can be discussed in isolation.

First, there is an interval of negative compression starting at about -5 % where a single out of 60 experiment runs is located. In this case, rather than reducing the number of rules in the advisor model, the application of classifier adaptation rather effectuates an increase of rules. The working hypothesis here is that the advice acquired from the advisor panel throughout the adaptation process did

not lead to a reconfiguration of the original advisee rule set but rather led to the addition of additional rules that specifically covered problematic classification cases in the advisee training set which have consequently been turned into subjects of advisory interaction steps.

Second, there are three simulation runs where neither an inflation nor a deflation of the number of rules in the advisee model has been measured. While no change in terms of the key performance indicator at hand suggests a lack of change in the model, this is not necessarily the case and the indirect observation should hence been analysed further. Considering the data points, i.e., experiments, that are lined up on the plot x-axis at 0 % compression, there are those where no error reduction has been achieved as well. In these cases, the classification model refinement process has been terminated immediately without effect. The advisor panel has been ineffective given the original advisee model. A potential explanation is that this model was possibly induced by a particularly helpful set of training instances.

As a side node, one can argue that the fraction of simulation runs that falls into this aforementioned group constitutes one possible indicator for the effectiveness of the available advisor panel and the domain *expertise* encoded in the respective classification models.

Turning the attention back to the experiment evaluation, the remaining experiments at 0 % model compression necessarily exhibit a mixture of advice-induced rule set reconfiguration with changes towards better generalisation (i.e., fewer rules) and addition of novel rules.

Finally, the same applies for the by far largest third group of experiment runs whose data points are located in the upper right sector of the plot in Figure 7.12. What sets these runs apart is a stronger correlation of error reduction for classification and signification reduction in rule set size.

**Finding**

The data supports the conclusion that the FARS 2011 Motorist data serves as a positive basis in order to illustrate desired adaptation effects. In particular, interaction on learning can effectuate in some cases a model compression of up to about 20 % ignoring outliers.

This is relevant for the following reason. The correlation plots show that by and large, rule set compression can translate to increased generalisation of the resulting model which then translates to less errors in future classification tasks.

### Analysis of Search-driven Adaptation Control Implementation

The evaluation hitherto discussed characteristics of the adapted classification models created in an adaptation episode. As a complement to these analyses, the behaviour of the implemented strategy that drives the control scheme of the adaptation process must be considered. Section 5.3 has modelled the adaptation of individual classification models as an online search problem. Consequently, Section 6.3.1 elaborated that the concrete implementation of the online search has been realised as an elementary hill climbing search. In doing so a baseline

| advisor panel | performance indicator | | mean | stddev | min | q25 | median | q75 | max | samples |
|---|---|---|---|---|---|---|---|---|---|---|
| Experimental basis: 60 simulation runs / data | | | | | | | | | | |
| Group 1: heterogeneous composition of advisor induction algorithms | | | | | | | | | | |
| 2xJ48+2xABCN2+2xRIPPER | *search depth* | # | 3 | 1.12 | 1 | 2 | 3 | 4 | 6 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 17.12% | 8.24% | 3.57% | 10.53% | 16.90% | 21.05% | 39.62% | 60 |
| | step #2 \|++ | | 14.48% | 9.40% | 2.04% | 6.90% | 12.82% | 18.92% | 39.39% | 55 |
| | step #3 \|+++ | | 12.29% | 8.78% | 2.33% | 5.26% | 10.34% | 15.79% | 34.15% | 40 |
| | step #4 \|++++ | | 6.37% | 4.76% | 1.96% | 2.78% | 4.76% | 6.67% | 17.14% | 20 |
| | step #5 \|+++++ | | 5.53% | 2.10% | 3.23% | 3.23% | 6.06% | 7.32% | 7.32% | 3 |
| | step #6 \|++++++ | | 2.71% | 0.21% | 2.56% | 2.56% | 2.56% | 2.86% | 2.86% | 2 |
| 3xABCN2+3xRIPPER | *search depth* | # | 3.133 | 1.171 | 1 | 2 | 3 | 4 | 7 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 17.57% | 7.81% | 5.26% | 12.00% | 16.28% | 21.43% | 37.31% | 60 |
| | step #2 \|++ | | 16.08% | 10.78% | 2.63% | 7.32% | 14.29% | 19.05% | 52.78% | 56 |
| | step #3 \|+++ | | 10.27% | 7.85% | 2.17% | 3.13% | 7.41% | 13.33% | 34.29% | 44 |
| | step #4 \|++++ | | 8.07% | 6.31% | 2.17% | 3.57% | 5.26% | 10.34% | 21.88% | 20 |
| | step #5 \|+++++ | | 10.02% | 6.06% | 2.86% | 2.86% | 7.14% | 14.71% | 15.38% | 4 |
| | step #6 \|++++++ | | 10.20% | 7.27% | 2.86% | 2.86% | 10.34% | 17.39% | 17.39% | 3 |
| | step #7 \|+++++++ | | 9.09% | | 9.09% | 9.09% | 9.09% | 9.09% | 9.09% | 1 |
| 3xJ48+3xRIPPER | *search depth* | # | 2.817 | 0.965 | 0 | 2 | 3 | 4 | 4 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 14.87% | 6.76% | 2.22% | 10.00% | 14.04% | 19.30% | 33.33% | 59 |
| | step #2 \|++ | | 13.89% | 9.78% | 2.56% | 5.41% | 11.90% | 20.00% | 42.86% | 55 |
| | step #3 \|+++ | | 8.19% | 6.05% | 2.27% | 3.03% | 6.06% | 12.50% | 24.32% | 39 |
| | step #4 \|++++ | | 6.56% | 4.38% | 2.38% | 3.03% | 5.26% | 7.69% | 18.75% | 16 |
| 3xJ48+3xABCN2 | *search depth* | # | 3.167 | 1.122 | 1 | 2 | 3 | 4 | 6 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 18.31% | 8.80% | 5.26% | 10.34% | 17.39% | 22.22% | 45.28% | 60 |
| | step #2 \|++ | | 14.44% | 7.49% | 2.22% | 8.82% | 14.04% | 18.60% | 33.33% | 57 |
| | step #3 \|+++ | | 9.68% | 6.65% | 2.08% | 5.00% | 8.33% | 11.76% | 30.56% | 44 |
| | step #4 \|++++ | | 6.39% | 4.37% | 1.85% | 2.63% | 5.26% | 8.33% | 18.18% | 21 |
| | step #5 \|+++++ | | 2.71% | 0.14% | 2.50% | 2.63% | 2.78% | 2.78% | 2.86% | 5 |
| | step #6 \|++++++ | | 15.82% | 17.02% | 3.03% | 3.03% | 9.30% | 35.14% | 35.14% | 3 |
| Group 2: homogeneous composition of advisor induction algorithms | | | | | | | | | | |
| 6xABCN2 | *search depth* | # | 3.033 | 0.938 | 1 | 3 | 3 | 4 | 6 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 18.31% | 8.80% | 5.26% | 10.34% | 17.39% | 22.22% | 45.28% | 60 |
| | step #2 \|++ | | 13.29% | 7.66% | 2.44% | 8.33% | 11.63% | 17.07% | 36.11% | 55 |
| | step #3 \|+++ | | 7.86% | 5.65% | 2.13% | 3.23% | 6.25% | 8.82% | 26.83% | 48 |
| | step #4 \|++++ | | 4.20% | 1.89% | 1.96% | 2.86% | 3.33% | 6.06% | 7.89% | 17 |
| | step #5 \|+++++ | | 34.38% | | 34.38% | 34.38% | 34.38% | 34.38% | 34.38% | 1 |
| 6xJ48 | *search depth* | # | 2.983 | 1.049 | 1 | 2 | 3 | 4 | 6 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 15.25% | 6.80% | 2.27% | 10.00% | 14.29% | 18.87% | 32.08% | 60 |
| | step #2 \|++ | | 13.11% | 8.97% | 2.56% | 5.88% | 11.76% | 18.92% | 44.68% | 56 |
| | step #3 \|+++ | | 8.41% | 6.19% | 2.13% | 3.45% | 6.25% | 11.43% | 32.35% | 41 |
| | step #4 \|++++ | | 5.03% | 2.87% | 2.38% | 2.70% | 3.57% | 6.67% | 11.76% | 17 |
| | step #5 \|+++++ | | 3.56% | 1.17% | 2.63% | 2.63% | 3.03% | 3.33% | 5.26% | 4 |
| | step #6 \|++++++ | | 2.78% | | 2.78% | 2.78% | 2.78% | 2.78% | 2.78% | 1 |
| 6xRIPPER | *search depth* | # | 1.783 | 0.825 | 0 | 1 | 2 | 2 | 4 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 9.80% | 5.64% | 1.79% | 5.17% | 8.33% | 13.64% | 23.19% | 56 |
| | step #2 \|++ | | 11.87% | 7.12% | 3.57% | 5.88% | 10.53% | 16.13% | 33.33% | 41 |
| | step #3 \|+++ | | 15.43% | 15.85% | 4.55% | 6.67% | 7.14% | 19.23% | 52.94% | 9 |
| | step #4 \|++++ | | 13.04% | | 13.04% | 13.04% | 13.04% | 13.04% | 13.04% | 1 |

**Table 7.3:** Search-oriented overview of experiment series, providing details for search depths and branching.

has been established for evaluation while leaving room for optimisation. As the evaluation is to further understanding on actual search cycles, Table 7.3 accumulates measurements from search-oriented key performance indicators across experiment series with different advisor panels.

Considering the distribution of search depths in terms of the statistical means listed in the table, it is revealed the search traces are typically rather short with mean values $\mu_{\text{exp}} \in \langle\, 1.783, 3.167 \,\rangle$. The highest means have been measured for heterogeneous compositions of advisor induction algorithms, specifically for the `3xABCN2+3xRIPPER` ($\mu = 3.133$, $\sigma = 1.171$) and `3xJ48+3xABCN2` ($\mu = 3.167$, $\sigma = 1.122$) advisor panels.

These results are in line with the observation that for homogeneous advisor panels, the ABCN2 algorithm alone outperforms the competition ($\mu = 3.033$, $\sigma = 0.938$). What is interesting however, is the fact that the involvement of lower-performing induction algorithms for parts of the advisor panel nevertheless, if only slightly, increase the length of search traces in the mean.

This observation is confirmed by measurements for the fraction of improving branching options for the different advisor compositions. Table 7.3 breaks down the fractions of improving branching based on the steps during adaptation episodes. Considering the initial adaptation step, the table shows a means for improving branching of $\mu \in \langle\, 9.80; 18.31 \,\rangle$ percent for the heterogeneous advisor panels with ABCN2 ($\mu = 18.31\% \pm 8.8\%$) dominating J48 and RIPPER. The table also shows that heterogeneous advisor panels through the involvement of ABCN2 and J48 lead to a consistently hight fraction of improving branching with means $\mu \in \langle\, 14.87; 18.31 \,\rangle$ percent.

These results constitute an indication that induction algorithm heterogeneity in advisor agents is a factor that promotes a better exploration of the model search space. This positive effect also correlates with the observations from the performance-oriented key performance indicators, at least with respect to the advisee's own test data as reference (cf. Figure 7.5 on page 161).

The experiments have also shown that the overall branching factor throughout adaptation episodes is considerable initially with a level-off effect in later adaptation steps. However, this still leads to a costly exploration for each search step. Since expansions from the current search state are investigated whenever potentially helpful advice can be acquired from the advisor panel for a given learning problem, and each investigation involves the induction of a new ABCN2 classifier using a data basis extended by the new advice, it is rendered evident that effective expansion heuristics are a worthwhile topic for future research, to be addressed in Section 8.2.

### 7.4.3 Micro Evaluation of Selected Experiment Runs

To further understanding of the effects of classification model adaptation episodes as they proceed in distinct refinement steps, the evaluation proceeds with the investigation of single experiment runs. These runs are highlighted in Figure 7.14. The plot refers to the advisee test data set as reference frame, i.e., the micro evaluation adopts the perspective of the agent seeking to enhance its classifier. The discussion includes two successful experiment runs from the agent perspective (i.e., run #21 and #41). Also, it includes a paradigmatic run that can be classified as only slightly successful (run #50). In addition, resources such as plots which support the evaluation are included for each single experiment run and all

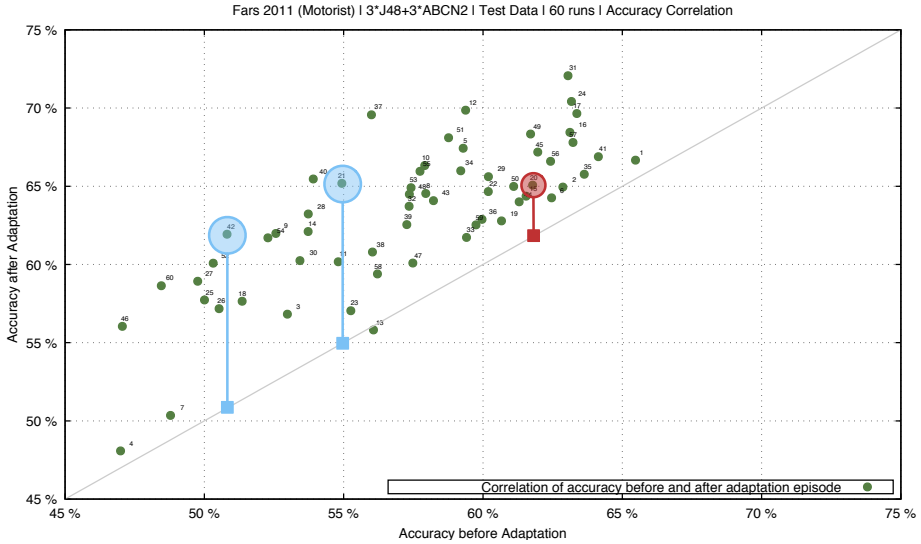



**Figure 7.14:** The correlation plot already known from Figure 7.10 on page 169 is used here in order to highlight experiment runs that are subject to the following micro evaluation.

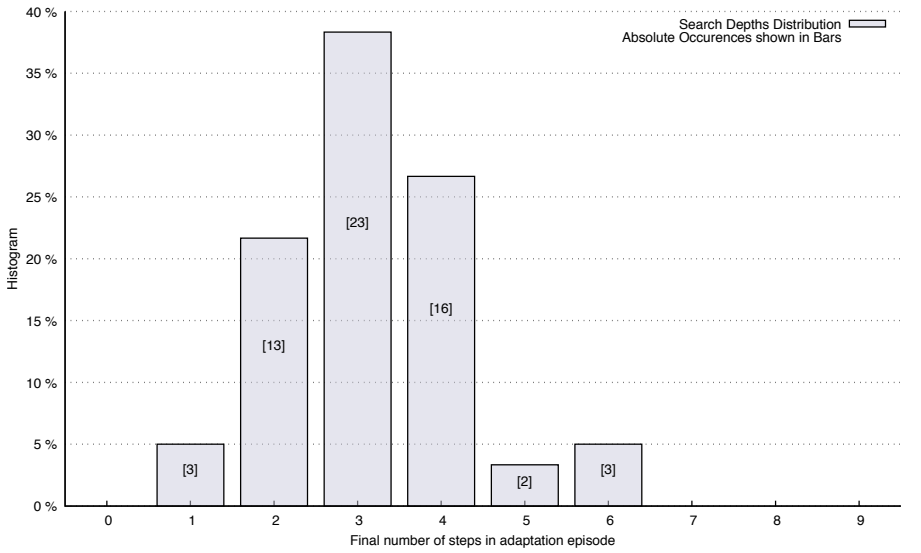



**Figure 7.15:** Context with a histogram of search depths for an experiment series.

experiment setups as part of the digital version of this thesis.

The treatment for each experiment run follows a common scheme. First, it starts with an overview plot of the scope and composition of the rule set that constitutes the advisee classifier over the course of an episode of interactive multiagent adaptation. A main plot tracks the classification model as it is refined with each successive adaptation step beginning with the original, unchanged rule set as an individual baseline. With this point of reference, the rule sets for subsequent steps are partitioned such that the individual numbers of persistent, i.e., essentially untouched rules, is given as a proportion, along with the body of

modified rules, deleted rules, and finally novel rules. The algorithm that enables the partition is a standard text diff algorithm which has been ported to work with sorted rule sets. Although the threshold between modified and novel rules is fuzzy, overview plots such as the one in Figure 7.24 provide an intuition into the rule set change characteristics induced for a given experiment setting and problem domain. As a compliment to the rule set modification, the step-wise adaptation is also quantified by means of average classification accuracy and Matthews Correlation Coefficient (cf. Figure 7.17).

Second, the adaptation changes are visualised precisely with confusion matrices. A first pair of these matrices shows a contrasting juxtaposition of the advisee classification result with the original classifier and its final refined successor. Additional difference matrices then break down the effects of the advice acquired in an adaptation step.

#### Experiment-Level Evaluation · `3xJ48+3xABCN2`, Run #21

A first experiment run chosen for detail evaluation is run #21 from the experiment series that used a mixed advisor panel of three ABCN2-enabled advisors and three using the J48 algorithm (`3xJ48+3xABCN2`, cf. Figure 7.14).

**Inspection of Model Evolution** Figure 7.16 illustrates that the investigated model refinement episode is composed of a series of four discrete refinement steps. The number of steps thereby corresponds to the depth of the search trace in the local search strategy (cf. Section 5.3). As can be read off the search depth histogram also presented in Figure 7.15, run #21 thus falls into the second-largest bin of experiment runs with 16 out of 60 experiments.

For model change behaviour, the data suggests that a large part of the model refinement occurs in the first two process steps. The number of items in the advisee rule set is immediately reduced by three rules and comprehensive change is detected in the remaining rules. Hence, the acquired advice in this first round of interaction has introduced a rule similar to those but not existing in the original model. This rule may have been subject to further refinement during the advisee's re-learning phase but ultimately let to the rule set reconfiguration and reduction. In the second step of the adaptation episode, the situation is repeated as new advice and subsequent re-learning allows to remove two more rules. In the remaining two steps, no further reduction in the number of rules is reached. However, the rule base is still further modified to a large degree induced by further adaptation. Figure 7.24 confirms the continuation of the initial adaptation scope.

**Evolution of Classifier Performance** The plot of the development of average predictive accuracy over the course of the adaptation episode and its steps in Figure 7.17 supplements the data presented in Figure 7.16 and the analysis presented thus far. The performance increases monotonously across all steps.

The comparison of the advisee measurements with those from the agents in the acting advisor panel is essential to put any performance gains discussed so far in context. In this regard, the results are encouraging insofar as the advisee

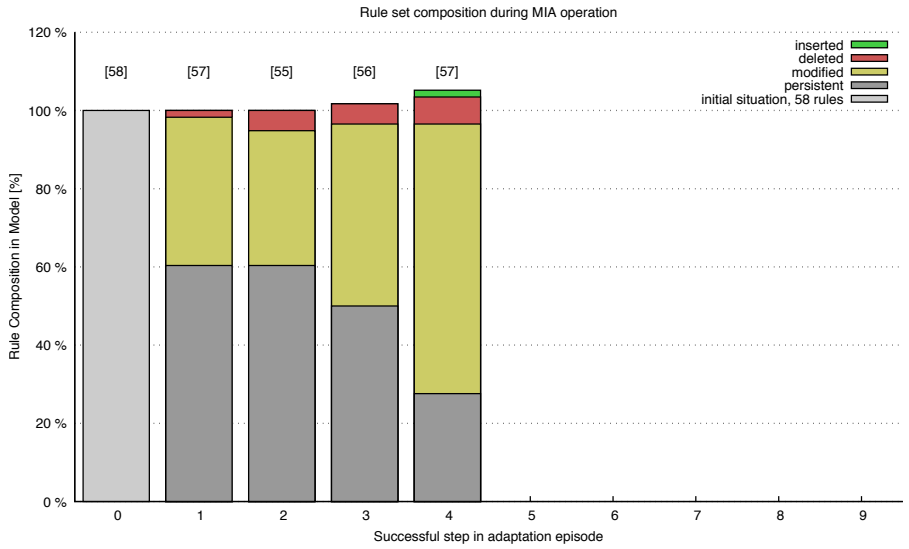
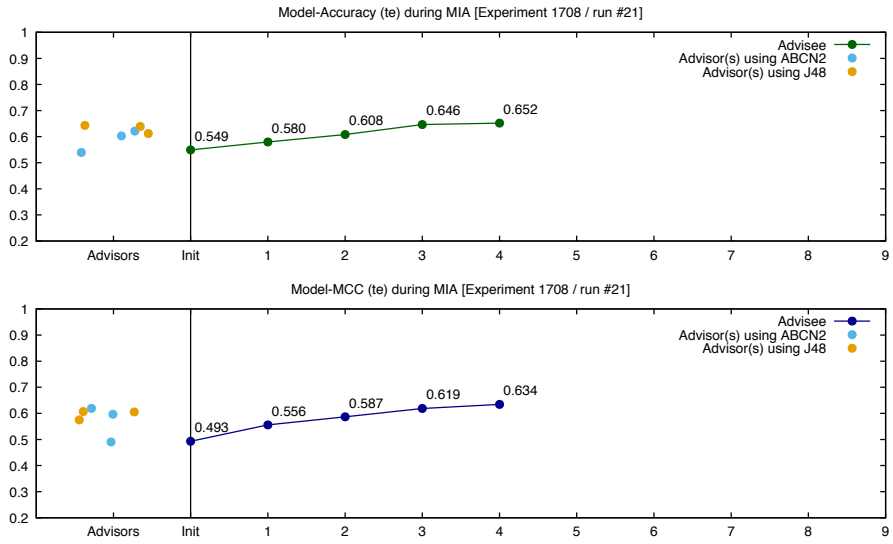



**Figure 7.16:** A plot with a detail analysis for a single experiment run from the experiment series `3xJ48+3xABCN2` (run #21). It breaks dow the modification of an advisee model during an adaptation episode. To quantify the change of the advisee rule set, the rules for different interaction steps have been ordered and compared by means of a diff-algorithm, hence coarsely highlighting rule change, rule removal and rule generation.

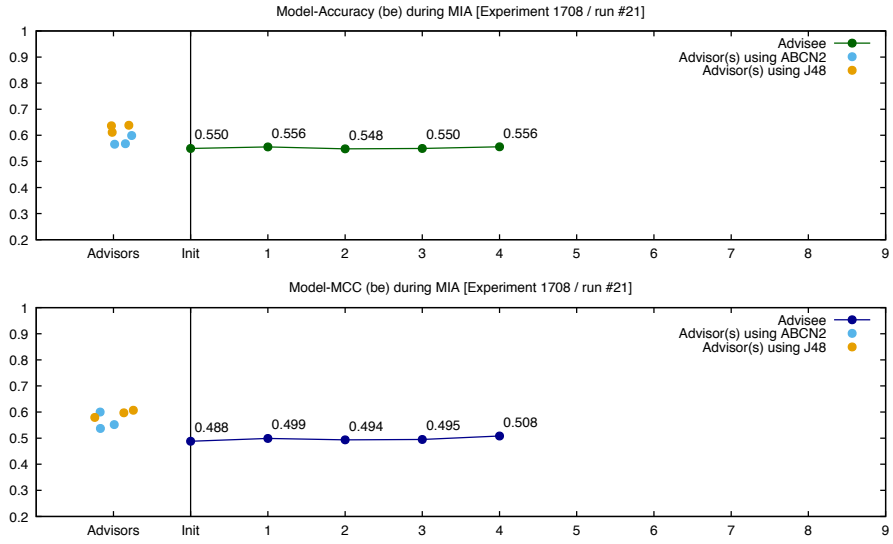
succeeds in raising itself to the level of the strongest ABCN2- and J48-powered advisors. The contrasting juxtaposition also offers a plausible explanation for the termination of the adaptation episode after four steps. More capable advisors may be required in the advisor panel to boost the advisee model beyond 65.2 % average classification accuracy.

A sampling inspection of other experiment runs, also across different experimental settings, reveals that a situation such as the one found in Figure 7.17 is a permanent trait for the investigated version of interactive multiagent adaptation. What can be learned from this part of the graphical analysis is that the approach does allow to refine the advisee model so as to be competitive with advisors. As such, it often constitutes a suitable instrument that allows for a close approximation of 'expert' advisors.

Hence, this state of affairs leaves room for improvement and ideas concerning which steps must be taken to enable the advisee agent in a typical adaptation scenario to tap the full potential of the best advisor agents more effectively. One conceivable scenario, discussed in Section 8.2 involves the inversion of control with respect to the choice of critical classification cases that are subject of advice acquisition. While these have been derived from the small-scale advisee training data alone, in future iterations, advisors could take a more active role by offering instances themselves. In knowledge management terms, the learning scenario would be shifted from the active, independent learner to a more classical classroom scenario.




**(a)** Advisee/advisor test data set frame of reference

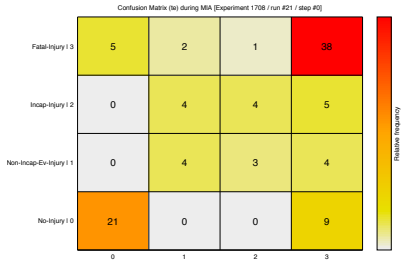



**(b)** Global benchmark data set frame of reference

**Figure 7.17:** Development of the advisee rule induction performance as measured with respect to accuracy (top plot) and Matthews Correlation Coefficient (cf. Section 5.3.1, bottom plot) against the test data set. The latter has been used in the examined experiment setup as the performance indicator that informed search process. The invariant advisor performance measurements are rendered on the left respectively.

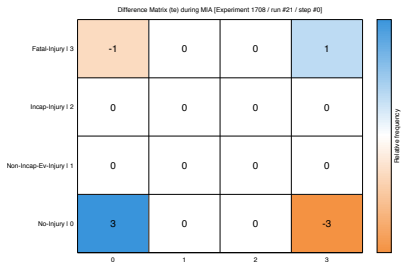
**Result Visualisation by Means of Confusion Matrices** As the detail evaluation of the first experiment run already offered new insights into the classifier refinement process and issues therein, this line of analysis is carried forward. The focus now turns on the discussion of confusion matrices beginning with those for the advisee test set.

Figure 7.18a depicts the initial confusion matrix measured against the advisee test data set. The matrix reveals several issues to be addressed by the inves-

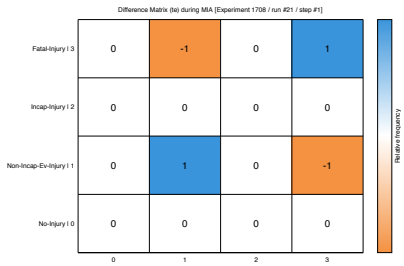



**(a)** Initial situation.

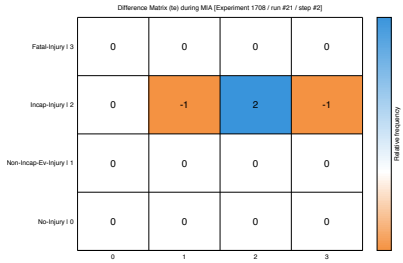

**(b)** Final situation.



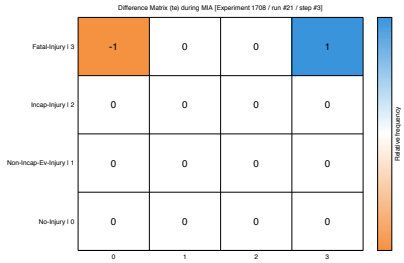

**(c)** Adaptation Step #1.



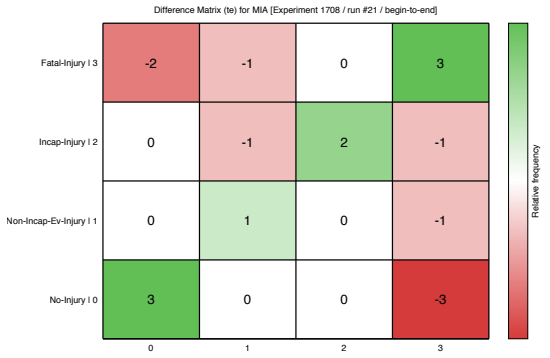

**(d)** Adaptation Step #2.




**(e)** Adaptation Step #3.




**(f)** Adaptation Step #4.




**(g)** Cumulative difference matrix for adaptation steps #1-#4

**Figure 7.18:** Visualisation of the model adaptation process for experiment run #21 in the `3xJ48+3xABCN2` experiment series (cf. Figure 7.8 on page 167). The first row of matrices directly contrasts the test set result obtained with the initial classifier with the result for the final modified version. The four matrices below constitute difference matrices that show the effects of the respective steps within the adaptation episode. The final matrix accumulates these discrete steps.

tigated model adaptation. First, the differentiation between the two majority classes `Fatal-Injury` and `No-Injury` is fuzzy with considerable misclassifications into the opposing class. Second, the two minority classes `Incap-Injury` and `Non-Incap-Ev-Injury` both show a common misclassification pattern with instances distributed relatively evenly across all classes expect `No-Injury`. Hence, the classifier rule base has issues to differentiate injury severities, specifically with the given domain.

Figure 7.18b consequently depicts the resulting confusion matrix after a completed adaptation episode. The overall matrix configuration has been retained throughout. This indicates that neither of the two issues raised in the previous paragraph has been addressed exhaustively. The difference matrix in Figure 7.18g provides a more focussed picture of the actual end-to-end changes. It is shown that the adaptation episode had a positive effect across all four domain target classes. This constitutes a best case scenario since adaptation has not focussed on a specific matrix region. However, in absolute terms, the number of instances that have been pushed onto the main diagonal of the matrix and are hence now correctly classified is outweighed by the remaining number of misclassifications. This can be explained by the advisor configuration (cf. Figure 7.17 on page 180) and potentially insufficient discriminatory power of the given domain features.

Figures 7.18c-7.18f display the discrete intermediate steps that cumulate in the difference matrix in Figure 7.18g. Step #1 focusses on an improved discrimination of the minority classes based on a `No-Injury` instance as learning problem. Step #2 involves a `Non-Incap-Ev-Injury` instance but in the process improves on the injury severity discrimination. Step #3 involves an `Incap-Injury` instance and resolves misclassification in the other two injury severity classes. Finally, step #4 addresses a `Fatal-Injury` instance that was originally misclassified as `No-Injury`. The step-wise breakdown of adaptation progress reveals several change patterns.

a) joint false-positive/false-negative resolution (Steps #1-2)
b) false-negative resolution (Steps #3-4).

Also, the effect of the adaptation steps is narrow in terms of the affected instances. Hence, while the advice procured by the advisor panel is adequate to advisee-defined problem instances, its incorporation as background knowledge for re-learning does not introduce seminal new induction rules with a broad effect measurable in terms of the confusion matrices. As evidence is presented that supports the assumption that the adaptation focusses on small regions in the space defined by the domain feature vector, a possible explanation for issue of reproducing adaptation success in the operationalisation phase is presented.

In order to substantiate these observations and respective derivations, further experiment samples are investigated in condensed form.

#### Experiment-Level Evaluation · `3xJ48+3xABCN2` #42

The second experiment run selected for a detail evaluation is run #42. It also belongs to the `3xJ48+3xABCN2` experiment series (cf. Figure 7.14).

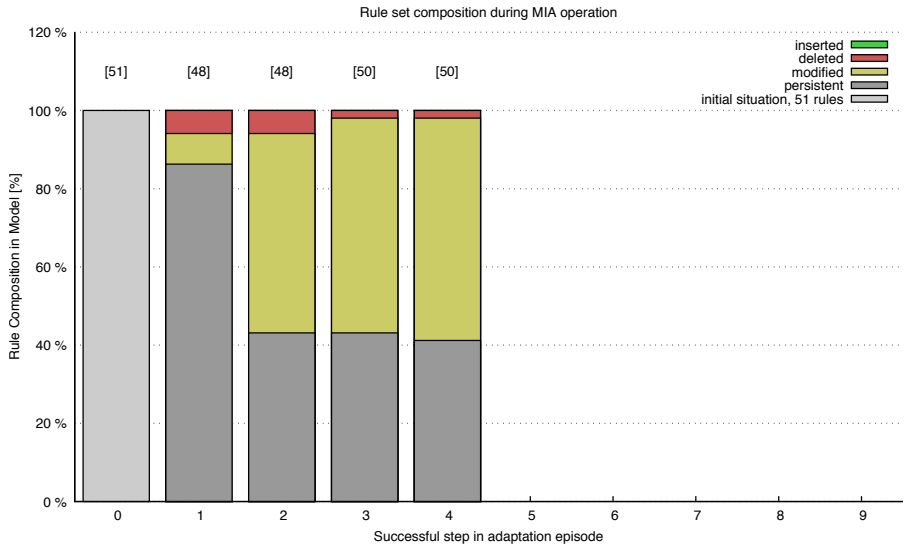
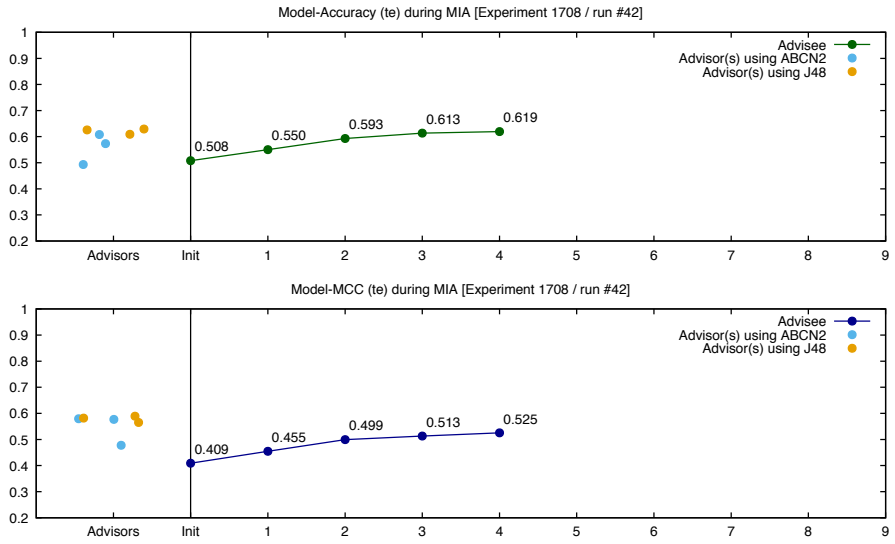



**Figure 7.19:** Detail analysis for a single experiment run from the experiment series `3xJ48+3xABCN2` (run #42). The upper part of the multi-plot provides a breakdown of the modification of an advisee model during an adaptation episode. To quantify the change of the advisee rule set, the rules for different interaction steps have been ordered and compared by means of a diff-algorithm, hence coarsely highlighting rule change, rule removal and rule generation. The lower plot contextualises the detail examination with a histogram of search depths for an experiment series.

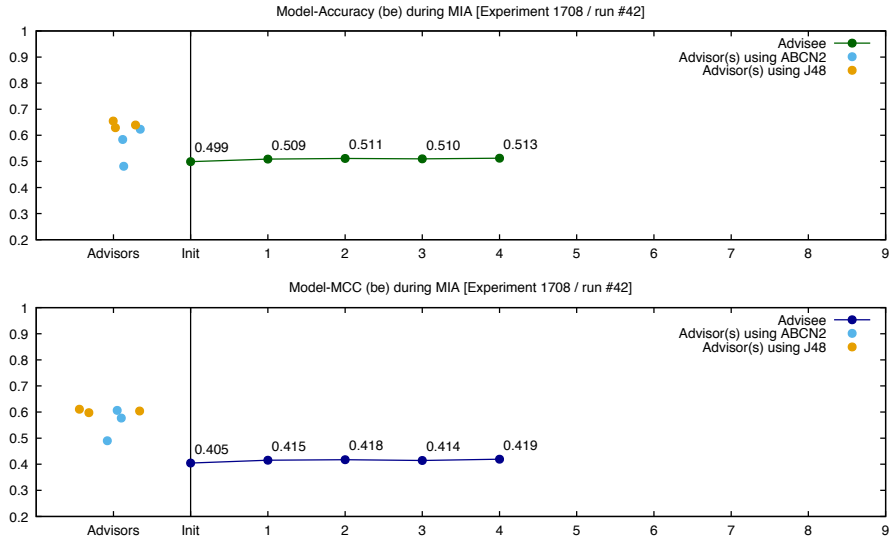
**Inspection of Model Evolution** Figure 7.19 illustrates that run #42 features a similar development of its classification model as run #21 discussed before, although the deviation in the absolute number of items in the classifier rule set is less pronounced. Starting with 50 rules, the first adaptation step eliminates two rules with a very limited change effect on the remaining rules. Step #2 then keeps the number of rules constant but evokes a large proportion of rule changes. According to Figure 7.20 this results in another notable performance improvement. The final two adaptation steps, by contrast, add another rule and do barely change additional parts of the rule base. In terms of the key performance indicators in Figure 7.20, this amounts to a level-off effect in the performance evolution graphs.

**Evolution of Classifier Performance** Figure 7.20 also confirms the measurements from experiment run #21 in that the advisee starts out in a situation where it judges its performance based on the given key performance indicators level or slightly below the weakest advisor. The observed adaptation process then propels it in its own assessment onto the level of the leading group of advisors.

**Result Visualisation by Means of Confusion Matrices** Figure 7.21 offers additional insight into the effects of the discrete adaptation steps in run #42 of the `3xJ48+3xABCN2` experiment series. The initial situation depicted in Figure 7.21a is such that the '`No-Injury`' instances in the advisee test set are already perfectly classified (i.e., no false-negatives). However, the remaining three target classes that represent degrees of accident-induced injuries are not differentiated




**(a)** Advisee/advisor test data set frame of reference



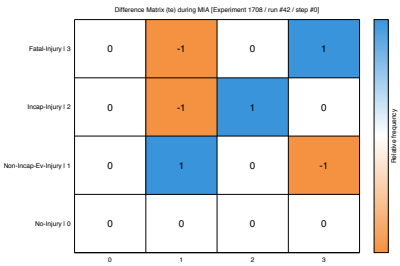

**(b)** Global benchmark data set frame of reference

**Figure 7.20:** Development of the advisee rule induction performance as measured with respect to accuracy (top plot) and Matthews Correlation Coefficient (cf. Section 5.3.1, bottom plot) against the test data set.

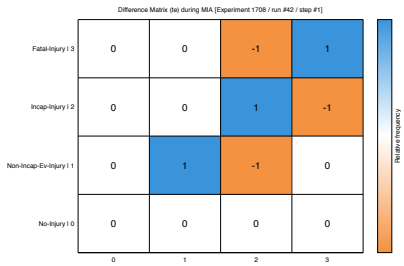
well. This is particularly true for the two minority classes '`Incap-Injury`' and '`Non-Incap-Ev-Injury`'. Consequently, model adaptation in this case is to concentrate on a monotonous improvement of the latter injury types while not worsening the situation in the '`No-Injury`' case. Indeed, Figure 7.21b confirms the desired behaviour. This is stressed even more in the difference matrix in Figure 7.21g. While an improvement can be stated for all desired target classes, the percentaged scope of these changes remains limited. This finding confirms the results seen previously for experiment run #21.


**(a)** Initial situation.

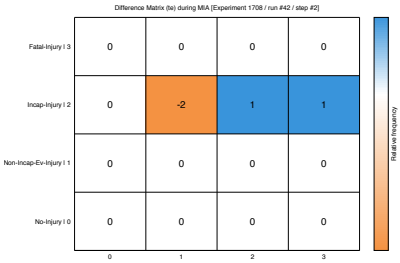

**(b)** Final situation.



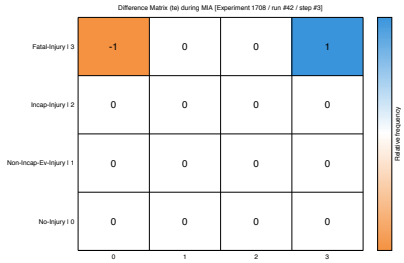

**(c)** Adaptation Step #1.



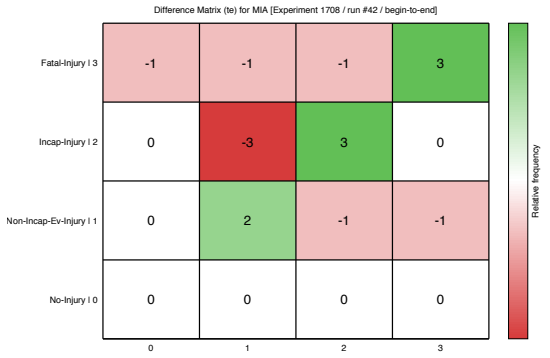

**(d)** Adaptation Step #2.




**(e)** Adaptation Step #3.




**(f)** Adaptation Step #4.




**(g)** Cumulative difference matrix for adaptation steps #1-#4

**Figure 7.21:** Visualisation of the model adaptation process for experiment #42 in the `3xJ48+3xABCN2` experiment series (cf. Figure 7.8 on page 167). The first row of matrices directly contrasts the test set result obtained with the initial classifier with the result for the final modified version. The four matrices below constitute difference matrices that show the effects of the respective steps within the adaptation episode. The final matrix consequently accumulates these discrete steps.

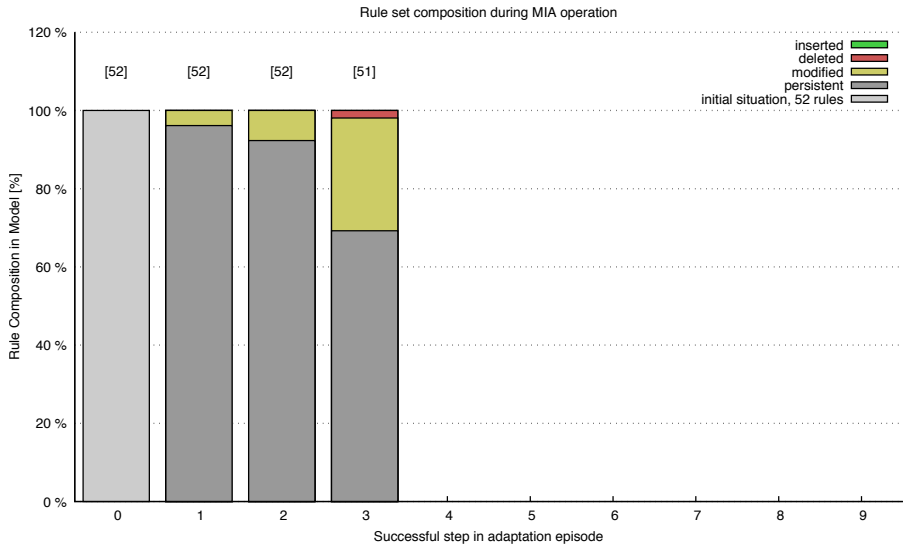


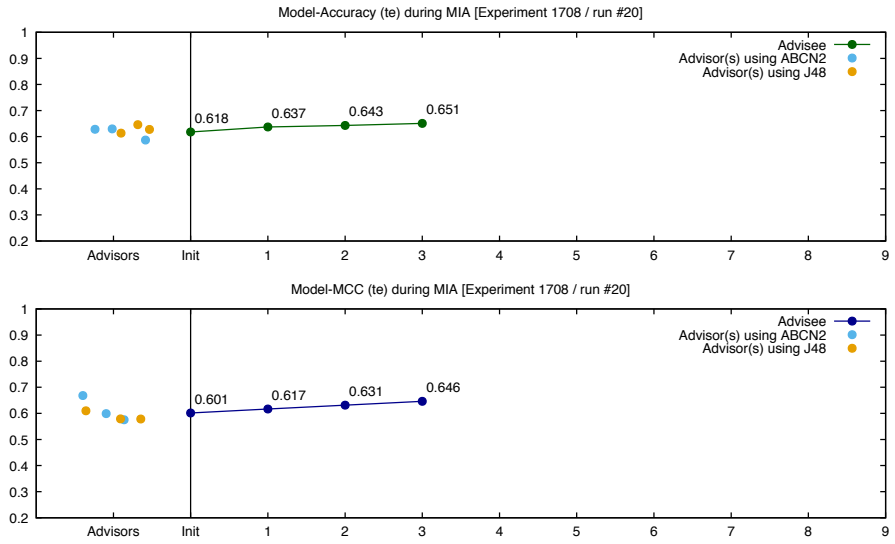

**Figure 7.22:** Detail analysis for a single experiment run from the experiment series `3xJ48+3xABCN2` (run #20). The upper part of the multi-plot provides a breakdown of the modification of an advisee model during an adaptation episode. To quantify the change of the advisee rule set, the rules for different interaction steps have been ordered and compared by means of a diff-algorithm, hence coarsely highlighting rule change, rule removal and rule generation. The lower plot contextualises the detail examination with a histogram of search depths for an experiment series.

With respect to the difference matrices, the first two adaptation steps (cf. Figure 7.21c and Figure 7.21d) address a '`Non-Incap-Ev-Injury`' and '`Incap-Injury`' instance respectively. The advice for both learning problems effectuates an improved discrimination equally spread across injury severities. These steps constitute paradigmatic examples for desirable adaptation pattern. The last two adaptation steps, which correlate with a levelling off of the adaptation progress (cf. Figure 7.20), address another '`Non-Incap-Ev-Injury`' learning problem and finally a fatal injury. Step #3 is thereby interesting as it improves the classification of '`Incap-Injury`' instances. Hence, the re-learning of the classifier based on the extended repertoire of learning advice leads to a desirable side-effect rather than addressing the original learning problem directly.

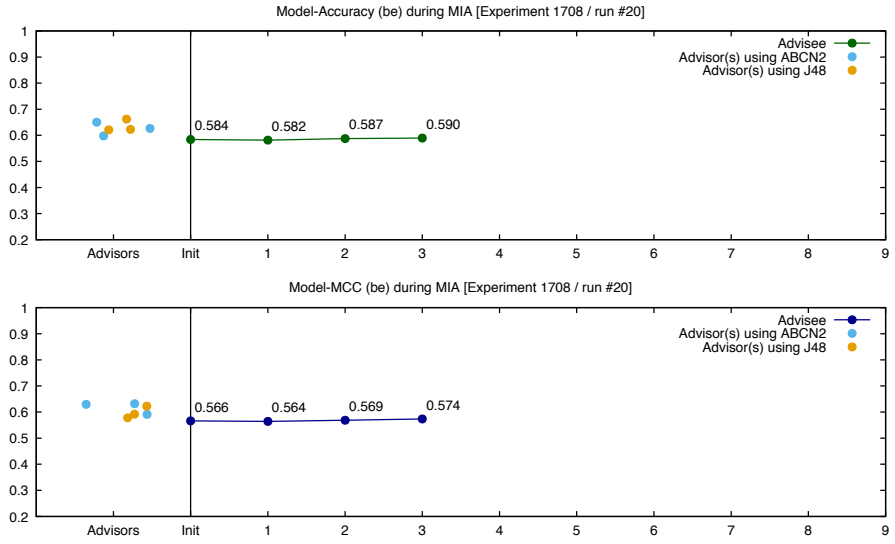
### Experiment-Level Evaluation · `3xJ48+3xABCN2` #20

The third experiment run selected for a detail evaluation is run #20. It also belongs to the `3xJ48+3xABCN2` experiment series. However, as Figure 7.14 highlights, this is an experiment run where the positive effect of the classifier adaptation episode is only about half as pronounced as in the experiment runs discussed this far.

**Inspection of Model Evolution** Consequently, the model evolution shown in Figure 7.22 documents a deviation from previous experiments. First, the search trace comprises only three discrete steps. Second, in the first two steps, the number of entries in the classifier rule set remains constant while the existing rules are also only varied to a minimal degree. As a consequence, respective




**(a)** Advisee/advisor test data set frame of reference




**(b)** Global benchmark data set frame of reference

**Figure 7.23:** Development of the advisee rule induction performance as measured with respect to accuracy (top plot) and Matthews Correlation Coefficient (cf. Section 5.3.1, bottom plot) against the test data set.

classifier performance improvements in terms of both the average classification accuracy and the Matthews Correlation Coefficient are minimal, as shown in Figure 7.23. Only in the last step does the model change pattern begin to resemble earlier experiments. To conclude, in this experiment run, the adaptation process is inhibited.

**Evolution of Classifier Performance** This observation is confirmed by the performance evolution plots in Figure 7.23 where the graphs are nearly flat,

starting from a better initial situation compared to experiment runs #42 and #21 (which corresponds to the x-axis position of run #20 in Figure 7.14). Due to the data sampling for this experiment run, including reservoir sampling by the warehouse agent and the training/test data split of the advisee, the original classifier of the advisee is graded better than in previous experiments. Then, over the course of the adaptation episode itself, significant further improvements cannot be effectuated with the given advisor panel. It is tempting to draw on the advisor measurements to form the assumption that weak overall capability of these interaction partners may have contributed to the weak adaptation performance. However, the examination of other experiment runs forbids this conclusion without substantiation by a dedicated investigation. Thus, the examination continues with the confusion matrices for further insights.

**Result Visualisation by Means of Confusion Matrices** The confusion matrix for the initial situation for experiment run #20, which is shown in Figure 7.24a on the facing page confirms the observation that with the advisee test data set as reference, the agent started with a comparatively well-trained classifier (cf. Figure 7.21 and Figure 7.24). As a consequence, the difference matrices for the adaptation steps in Figure 7.24 that add up to the cumulative matrix in Figure 7.24f only lead to micro-improvements. That is, each addressed learning problem for one of the target classes '`Non-Incap-Ev-Injury`' (Step #1), '`Fatal-Injury`' (Step #2), and finally '`No-Injury`' pushes a single previously misclassified false-negative instance in the test dataset onto the diagonal of correctly classified instances. Hence, the efficiency factor of the applied interactive multiagent adaptation in this observed experiment run trails the previous runs.

### 7.4.4 Discussion of Findings

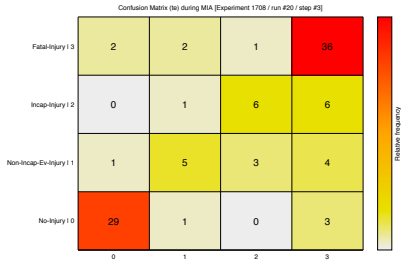
Following the conduct of both stages of the evaluation of the FARS 2011 Motorist dataset in Sections 7.4.2 and 7.4.3, the gist of findings can be presented.

Concluding the macro-evaluation for the FARS dataset, it can be stressed that notwithstanding the identified factor of instability (cf. Section 7.4.2 on page 166 ff.) the gist of the macroscopic evaluation in the FARS 2011 Motorist problem domain has confirmed the feasibility of a transfer of the ABCN2 approach of refining classification models based on human expert advisory into a multi-agent environment. In addition, points of optimisation have been identified at several areas in the model refinement framework. These areas are associated with well-defined extension points and hence can be addressed incrementally as part of future work without necessitating changes to the end-to-end integrated solution.

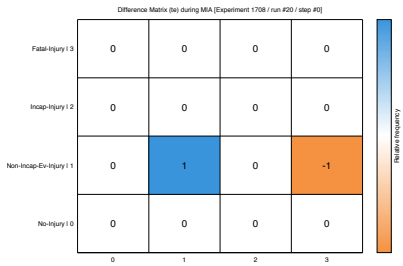
Concluding the micro-evaluation for the experiment runs #21, #42, and #20 it can be stated that a differentiated impression emerges for those experiment instances which have been seen as effective adaptation episodes from the advisee perspective. The principal functioning and potential of the investigated approach to multiagent interactive adaptation of individual classification models could be substantiated by example.

This complements likewise results discussed in the initial macro-evaluation for

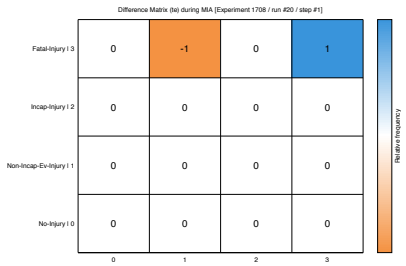

**(a)** Initial situation.



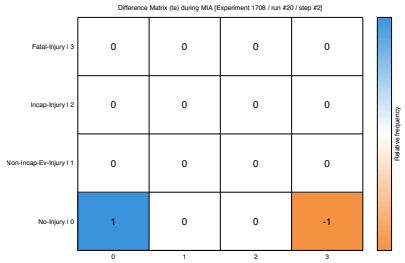

**(b)** Final situation.



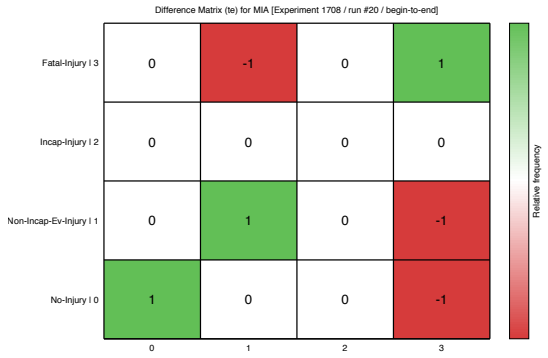

**(c)** Adaptation Step #1.




**(d)** Adaptation Step #2.




**(e)** Adaptation Step #3.




**(f)** Cumulative difference matrix for adaptation steps #1-#3

**Figure 7.24:** Visualisation of the model adaptation process for experiment #42 in the `3xJ48+3xABCN2` experiment series (cf. Figure 7.8 on page 167). The first row of matrices directly contrasts the test set result obtained with the initial classifier with the result for the final modified version. The four matrices below constitute difference matrices that show the effects of the respective steps within the adaptation episode. The final matrix consequently accumulates these discrete steps.

| Dataset | Source | Features | | | Samples | Classes |
|---|---|---|---|---|---|---|
| | | ∪ | Nominal | Numeric | | |
| Covertype Comanche | UCI | 12 | 2 (16.7%) | 10 (83.3%) | 80,000 | 6 |
| Covertype Neota | UCI | 12 | 2 (16.7%) | 10 (83.3%) | 29,884 | 3 |

**Table 7.4:** Overview of the datasets used in the experiment series for this evaluation. Additional information with respect to the distribution of classes for the target concept are presented in Figure 7.25.

the FARS 2011 Motorist dataset. However, the discussion of experiment runs has highlighted potential for optimisation. The graphical development of classifier performance (cf. Figure 7.17 and Figure 7.20) indicates a swift level-off and subsequent early termination of local searches in the model space.

The investigation of the complementary matrix plots further suggest that even in positive cases, adaptation improvements are focussed locally (quantitive matrix change behaviour documented in the difference matrices). This circumstance aids to explain the problems in retaining measured adaptation effects consistently for a subsequent classifier operationalisation.

## 7.5 Empirical Study on the Land Covertype Dataset

The second experiment series in this empirical study is based on data from the US Forest Service (USFS) Region 2 Resource Information System (RIS). Records within this dataset describe different types of forest cover for 30x30 meter patches of land. The complete dataset comprises 581,012 instances that are characterised by a total of 54 categorical and numeric features (cf. Section 5.1).

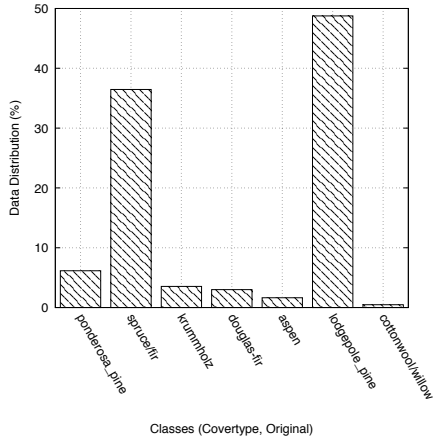
A normalised version of the dataset was used as a point of origin, obtained from the Massive Online Analysis (2016) repository.

### 7.5.1 Data Characteristics and Processing

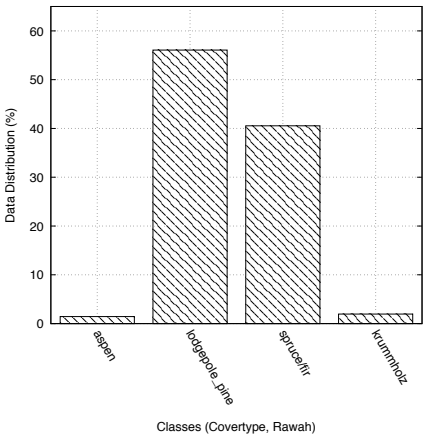
An inspection of the dataset revealed that it actually involved data from five geographically distinct regions within a large national park that varied significantly in topology and, as a consequence, land cover. Figure 7.25 illustrates this circumstance by enumeration the land covertype target classes and their distribution across all areas (Figure 7.25a) and the respective sub-area, i.e., Rawah, Comanche, Cache and Neota, in isolation.

Figure 7.25a immediately shows that the covertype target classes overall can be partitioned into extreme majority (`spruce/fir` and `lodgepole_pine`) and minority classes (`ponderosa_pine`, `krummholz`, `douglas-fir`, `aspen`, `cottonwood/willow`), some of them exclusive to certain sub-areas. The minority classes are thereby a natural consequence of the topology as for instance krummholz only grows right below the timberline in high altitudes.

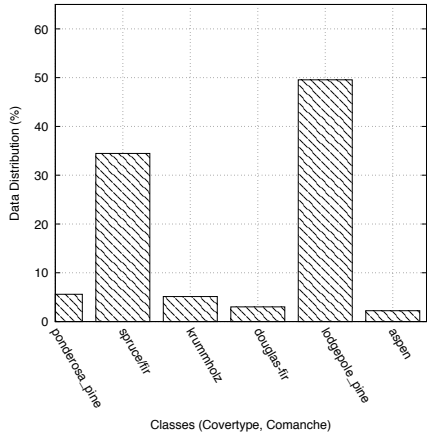
For the experiment series to be conducted, a focus was hence consciously set



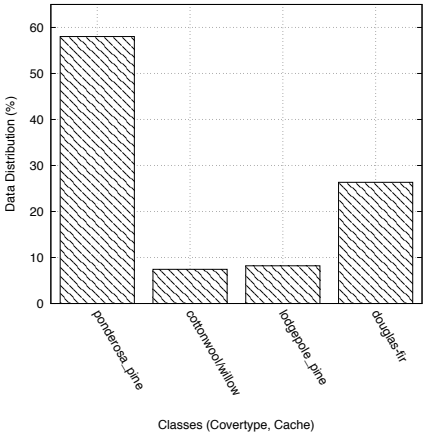

**(a)** Classes (Land covertype, all regions)



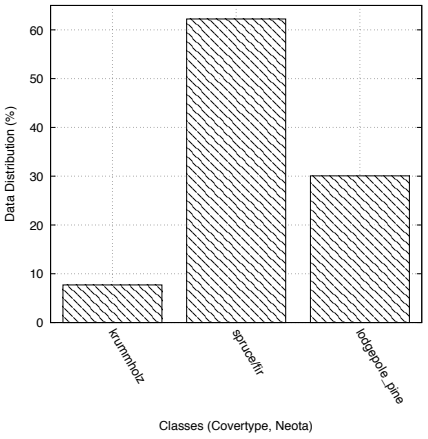

**(b)** Classes (Land covertype, Rawah region)




**(c)** Classes (Land covertype, Comanche region)




**(d)** Classes (Land covertype, Cache region)




**(e)** Classes (Land covertype, Neota region)

**Figure 7.25:** Histogram plots which break down the class distribution for the target concept for the dataset used in the experiment series for the evaluation.

on the Neota dataset as it was less encumbered with class imbalances than for instance Rawah or Comanche. In addition, the preliminary study on regarding the performance evolution for growing numbers of training instances on all applied induction techniques (Figure 7.26 on the next page) suggested an interesting point of origin for the application of adaptation.

Figure 7.4 shows that the restriction to the Neota area confined the data repository that is available as basis for dataset sampling for all simulation agents to 29,884 instances. This was still sufficient to proceed with the experiment conduct.

### Data Distribution among Advisee and Advisors

Once the repository data for the land covertype dataset was uploaded to the experiment database, a suitable assignment of instances for the actors involved in the planned experiment series on the Neota sub-area had to be determined.

Analogous to the procedure for the FARS 2011 Motorist experiment series discussed in Section 7.4, an empirical pre-study has also been conducted which had learning agents using each of the classification schemes employed in the final study repeatedly with a growing sampling size of data randomly drawn from the global repository. In 60 repetitions for each data sampling size, the classifier performance on a very large reference data set of 20,000 samples was measured in terms of the Matthews Correlation Coefficient. The results of this pre-study are shown in the learning curves in Figure 7.26.

Based on the learning curve plots in this figure, a reasonable reading for the size of advisor and advisee has been taken. Specifically, the advisor data size is determined by the point in the respective curve where growth essentially levels off. This indicates, that a further enlargement of the data set without further input augmentation efforts will demonstrably not yield better classifier performance. For the advisee, the data set size had to be chosen such that an inexperienced learner is modelled. Consequently, as shown in the top-most plot in Figure 7.26, a point in the learning curve with a steep inclination is chosen.

Section 7.4.1 on page 160 argued that besides data set sizing, the learning curves allow for the derivation of assumptions on the scope of the effect of interactive multiagent adaptation of classification models for a given task domain. To begin with, it can be stated that point of origin for the Neota experiments is different in several aspects. First, the learning curves are notably more diverse relative to the FARS versions. The learning curve for the ABCN2 learners shows a large variability for all sampling sizes. This may translate to similarly large variability in the effectiveness of the investigated adaptation approach. Interestingly, the RIPPER and J48 curves show a greater focus and also grow more aggressively for smaller sampling sizes before beginning to level-off on a much higher level ($mcc > 0.7$) than ABCN2 ($mcc \sim 0.55$). Therefore, by those initial numbers, the performance spread between advisee and top advisors is large and could thus effectuate larger gains in adaptation episodes than seen before in the FARS experiment series. Specifically, the learning curves support the assumption that advisors based on the RIPPER or J48 induction techniques may boost those experiments where they are involved.

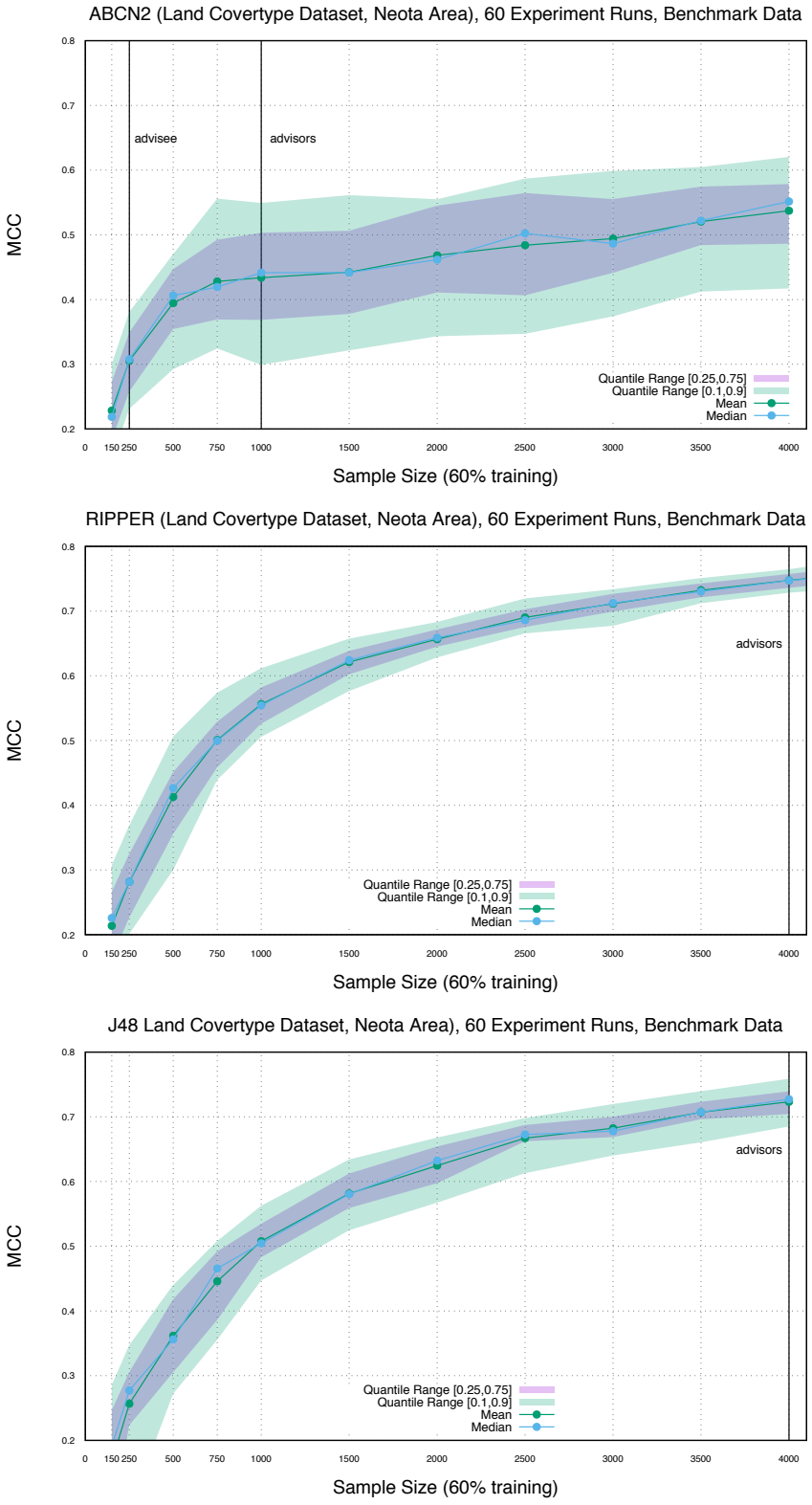



**Figure 7.26:** Learning curves based on 60 distinct experiments runs for each of the base classification algorithms that are employed by advisors (ABCN2, RIPPER, and J48) and advisees (ABCN2 only) in the experiment series for Neota lang covertype data set. The vertical lines denote the empirically determined number of instances to be procured to the respective agents in experiments.

### 7.5.2 Macro Evaluation of the Experiment Series

In the following, the performance and characteristic traits of interactive multiagent adaptation in an experiment series based on the Neota land covertype dataset are investigated from a high-level perspective.

As for the FARS experiment series, Figure 7.27 and Figure 7.28 cumulate a total of 420 distinct simulations spread across seven experiments. The left side of the plot comprises four experiments with different induction algorithm mixtures of the six heads advisor panel. This is denoted on the x-axis of the plots. One mixture covers three induction algorithms while the other constitute equal-sized mixtures of a subset of two algorithms. Finally, the right side of the plot shows results for each algorithm in a homogeneous advisor panel.

#### Observations on Experiments with Homogeneous Advisor Panels

With initial attention on those experiments that have been conducted with homogeneous advisor panels ('Group 1' in Figures 7.27, 7.28 and complementing Table 7.5), the performance measurements show a clear but unexpected ranking of experiments, given the learning curves in the experiment pre-study (cf. Figure 7.26 on page 193).

Best results have been recorded for the `6xABCN2` experiment. With the advisee test data frame of reference, the ABCN2-advisors effectuate a mean delta in average classification accuracy of $\mu_{\Delta(acc)} = 8.62 \pm 6.21\,\%$ which amounts to an error reduction of $\mu_{err} = 18.19 \pm 12.74\,\%$. The median delta accuracy resides at $7.27\,\%$ with an inter-quantile range of $\langle\, q_{10}, q_{90} \,\rangle_{\Delta(acc)} = \langle\, 1.53, 15.83 \,\rangle\,\%$ (cf. Table 7.5a).

The `6xJ48` experiment follows with a mean delta in average classification accuracy of $\mu_{\Delta(acc)} = 7.10 \pm 5.78\,\%$ which amounts to an error reduction of $\mu_{err} = 15.18 \pm 12.33\,\%$. The median delta accuracy resides at $5.44\,\%$ with an inter-quantile range of $\langle\, q_{10}, q_{90} \,\rangle_{\Delta(acc)} = \langle\, 0.67, 13.84 \,\rangle\,\%$.

Finally, the weakest result has been reached in the `6xRIPPER` experiment. The numbers here are as follows. $\mu_{\Delta(acc)} = 5.17 \pm 4.03\,\%$ with an error reduction of $\mu_{err} = 11.31 \pm 9.17\,\%$. The median delta accuracy resides at $4.72\,\%$ with an inter-quantile range of $\langle\, q_{10}, q_{90} \,\rangle_{\Delta(acc)} = \langle\, 0.56, 1147.84 \,\rangle\,\%$.

**Finding**

Counter-intuitive at first, the recorded data for the homogeneous Neota land covertype experiments shows a ranking that is reciprocal to the performance measurements for respective learning agents in the pre-study. The results confirm that RIPPER and J48-based advisors have not been able to procure advice for the presented learning problems whose consideration as background knowledge in re-learning processes led to novel generalisations that boosted classifier performance beyond the level of the ABCN2 experiment.

Consequently, the experiments suggest that the utility of an advisor in the investigated implementation of interactive multiagent adaptation of individual classification models is not only dependent on the standalone performance of the advisor models but on the process of concrete advice synthesis for given learning

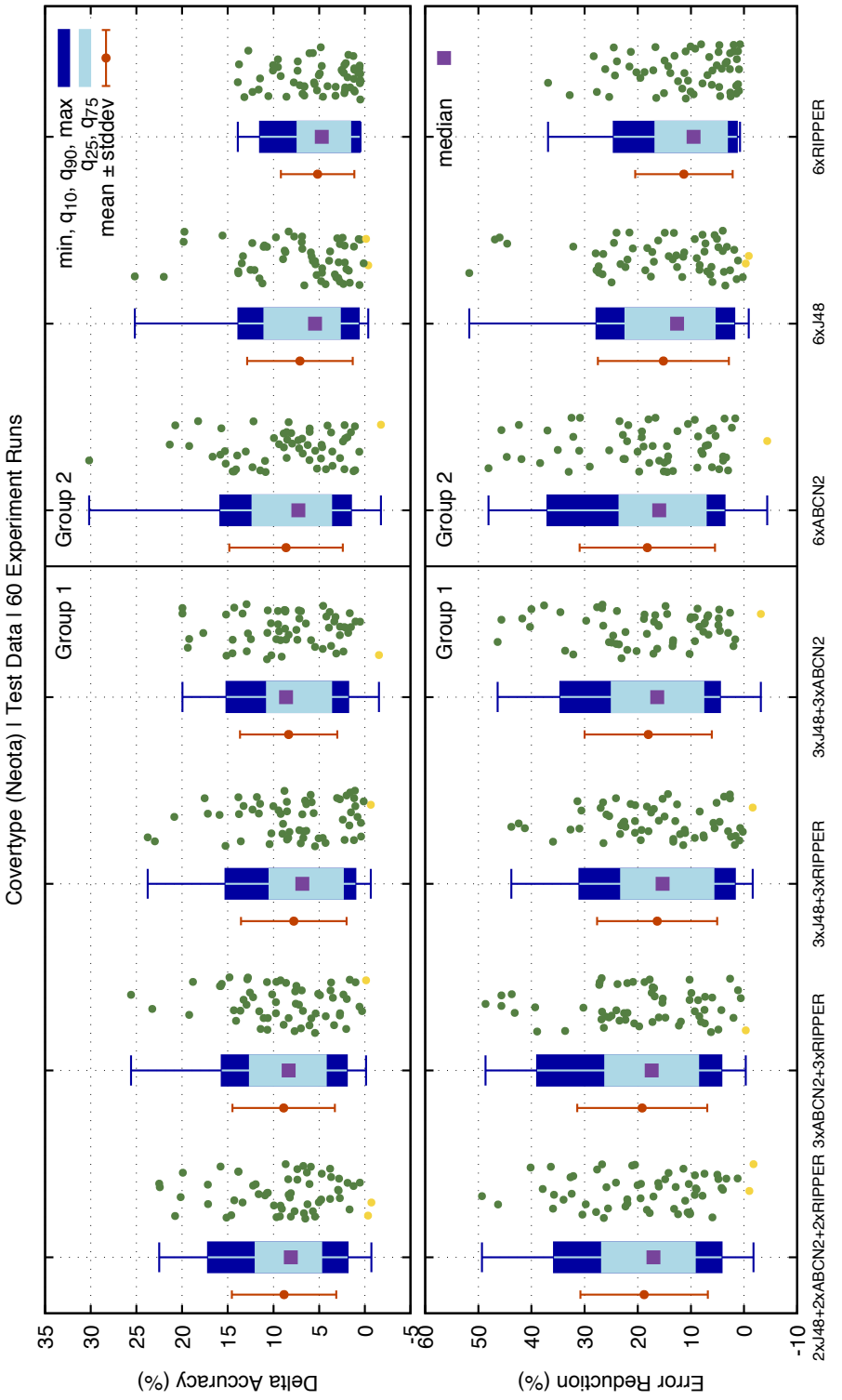



**Figure 7.27:** Comparative performance overview for multiagent interactive adaptation in an experiment series based on the Neota land covertype dataset. Each entry on the x-axis represents a distinct advisor composition in terms of the employed rule/decision tree learning algorithm with a fixed total of six advisors per experiment. Each experiment run contributes a single data point. The y-axis focuses on the adaptation-induced effect on classification performance with respect to average accuracy.

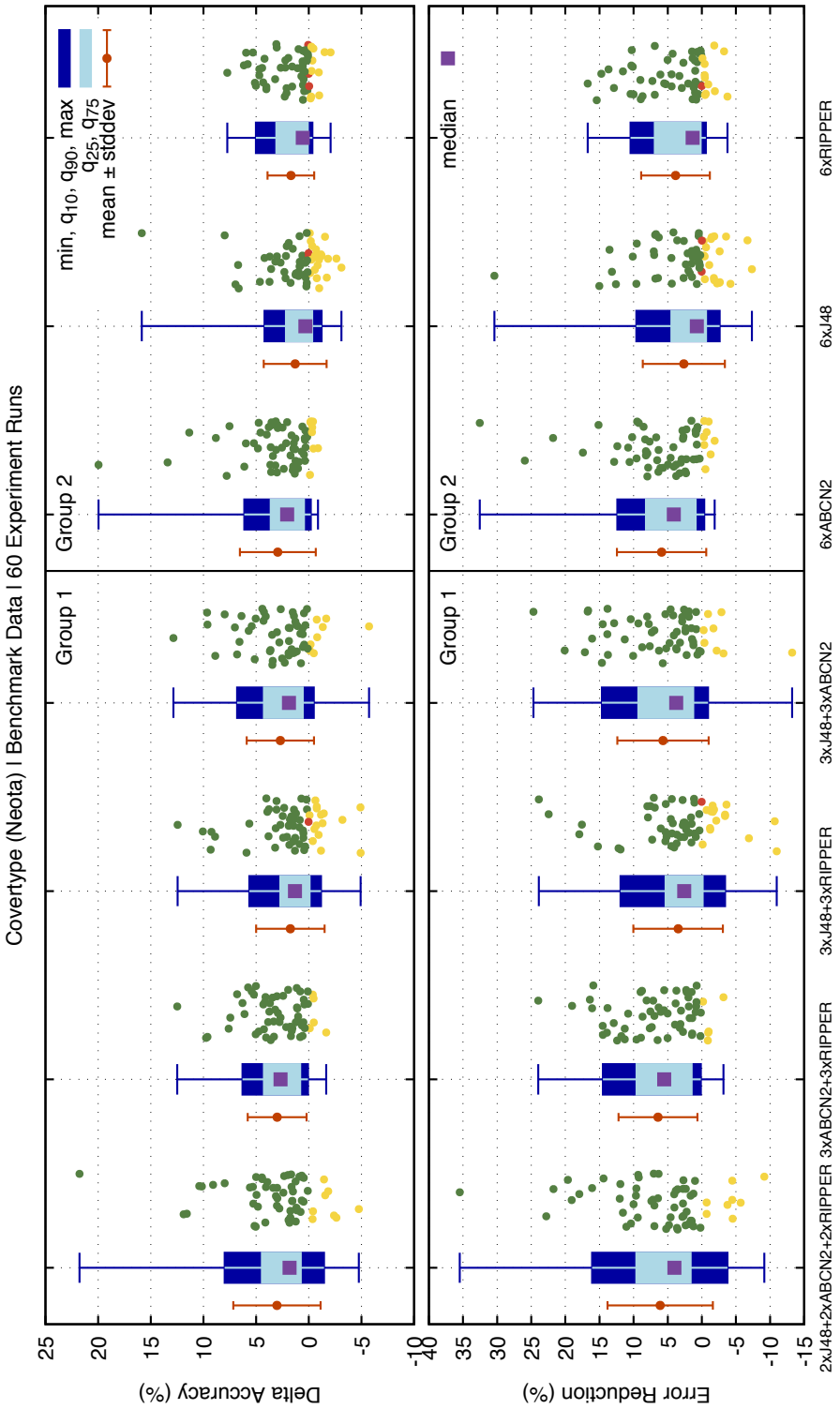



**Figure 7.28:** Results measured on the basis of a large-scale representative benchmark data set drawn independently from the complete universe of samples.

problems.

With respect to performance measurements based on the global benchmark dataset as reference, Figure 7.28 confirms, as expected based on the previous FARS 2011 Motorist investigation, that the performance across all three experiments is reduced relative to the test results above. While the `6xABCN2` experiment configuration still exhibits the best results in what could be seen as a simulated operationalisation of the adapted advisee classifier, the `6xJ48` and `6xRIPPER` experiments now exhibit comparable results.

The data for `6xABCN2` shows a mean delta in average classification accuracy of $\mu_{\Delta(acc)} = 2.93 \pm 3.6\,\%$ and an error reduction of $\mu_{err} = 5.9 \pm 6.55\,\%$. The median accuracy lies at $4.11\,\%$ with an inter-quantile range of $\langle\, q_{10}, q_{90} \,\rangle_{\Delta(acc)} = \langle\, -0.18, 6.12 \,\rangle\,\%$ or $\langle\, q_{25}, q_{75} \,\rangle_{\Delta(acc)} = \langle\, 0.45, 3.62 \,\rangle\,\%$.

**Finding**

For the most effective homogeneous experiment, in about 85 % of the time, an advisee that has completed an adaptation episode can expect to see a positive effect for future classifier operation. This effect can yield an absolute delta in average accuracy of up to 6 % disregarding outliers.

Compared with previous FARS 2011 Motorist results discussed in Section 7.4.2, the data in the macro-evaluation confirms that, from a global perspective, the interactive multiagent adaptation approach yields better results in the land covertype domain than in the accident domain discussed before. While the operationalisation of a modified classifier following an adaptation episode is still not guaranteed to improve performance, the probability thereof and the potential yield are in support of an application.

### Observations on Experiments with Heterogeneous Advisor Panels

Following the initial analysis for experiments with heterogeneous advisor panels, the focus is consequently put on the remaining heterogeneous settings ('Group 2' in Figures 7.27, 7.28 and complementing Table 7.5). The respective results are thereby discussed relative to the earlier homogeneous results as baseline.

Figure 7.27 shows several interesting facts for the investigated heterogeneous experiment configurations.

First, the `3xJ48+3xRIPPER` experiment, while still the weakest heterogeneous configuration, outmatches both homogeneous experiments for the constituent RIPPER and J48 advisor types with a mean delta in average classification accuracy of $\mu_{\Delta(acc)} = 7.78 \pm 5.78\,\%$, an error reduction of $\mu_{err} = 16.35 \pm 11.29\,\%$. The median delta accuracy is $6.84\,\%$ with a $\langle\, q_{10}, q_{90} \,\rangle_{\Delta(acc)} = \langle\, 1.04, 15.25 \,\rangle\,\%$ inter-quantile range.

For comparison, the numbers for the `3xJ48` experiments are: $\mu_{\Delta(acc)} = 7.1 \pm 5.78\,\%$, $\mu_{err} = 15.18 \pm 12.31\,\%$, a $5.44\,\%$ median and a $\langle\, q_{10}, q_{90} \,\rangle_{\Delta(acc)} = \langle\, 0.67, 13.84 \,\rangle\,\%$ inter-quantile range. As the effect also retained in the global benchmark context (cf. Table 7.5b for concrete measurements), this constitutes

| advisor panel | performance indicator | mean [%] | stddev [%] | min [%] | q25 [%] | median [%] | q75 [%] | max [%] |
|---|---|---|---|---|---|---|---|---|
| Data set: advisee-test, Experimental basis: 60 simulation runs | | | | | | | | |
| | *initial accuracy* | 52.422 | 9.377 | 33.698 | 46.834 | 51.547 | 58.608 | 70.868 |
| Group 1: heterogeneous composition of advisor induction algorithms | | | | | | | | |
| 2xJ48+2xABCN2+2xRIPPER | *final accuracy* | 61.263 | 9.856 | 40.873 | 55.37 | 60.714 | 67.016 | 80.415 |
| | *delta accuracy* | 8.841 | 5.718 | -0.729 | 4.779 | 8.095 | 11.945 | 22.504 |
| | *error reduction* | 18.807 | 11.994 | -1.792 | 9.273 | 17.048 | 26.713 | 49.308 |
| 3xABCN2+3xRIPPER | *final accuracy* | 61.305 | 10.565 | 38.889 | 52.282 | 62.172 | 68.688 | 79.224 |
| | *delta accuracy* | 8.883 | 5.601 | -0.139 | 4.287 | 8.333 | 12.569 | 25.596 |
| | *error reduction* | 19.162 | 12.241 | -0.341 | 8.652 | 17.396 | 26.131 | 48.654 |
| 3xJ48+3xRIPPER | *final accuracy* | 60.202 | 9.717 | 38.342 | 52.358 | 60.542 | 67.709 | 76.412 |
| | *delta accuracy* | 7.78 | 5.777 | -0.66 | 2.415 | 6.844 | 10.417 | 23.761 |
| | *error reduction* | 16.352 | 11.288 | -1.621 | 5.768 | 15.308 | 23.12 | 43.784 |
| 3xJ48+3xABCN2 | *final accuracy* | 60.767 | 10.472 | 38.889 | 52.282 | 60.44 | 69.747 | 77.869 |
| | *delta accuracy* | 8.345 | 5.32 | -1.554 | 3.681 | 8.624 | 10.71 | 19.964 |
| | *error reduction* | 18.023 | 11.969 | -3.154 | 7.679 | 16.339 | 24.892 | 46.378 |
| Group 2: homogeneous composition of advisor induction algorithms | | | | | | | | |
| 6xABCN2 | *final accuracy* | 61.038 | 10.208 | 37.238 | 54.685 | 62.759 | 67.426 | 78.767 |
| | *delta accuracy* | 8.616 | 6.214 | -1.771 | 3.681 | 7.272 | 12.299 | 30.176 |
| | *error reduction* | 18.191 | 12.744 | -4.352 | 7.238 | 15.95 | 23.415 | 48.089 |
| 6xJ48 | *final accuracy* | 59.525 | 10.48 | 38.889 | 52.096 | 59.167 | 66.484 | 77.622 |
| | *delta accuracy* | 7.103 | 5.778 | -0.371 | 2.727 | 5.443 | 10.994 | 25.185 |
| | *error reduction* | 15.182 | 12.331 | -0.896 | 5.527 | 12.582 | 22.303 | 51.693 |
| 6xRIPPER | *final accuracy* | 57.594 | 10.418 | 34.217 | 52.381 | 56.991 | 65.057 | 76.193 |
| | *delta accuracy* | 5.172 | 4.025 | 0.469 | 1.613 | 4.722 | 7.357 | 13.898 |
| | *error reduction* | 11.312 | 9.172 | 0.762 | 3.23 | 9.51 | 16.652 | 36.86 |

**(a)** Results measured based on the advisee test dataset.

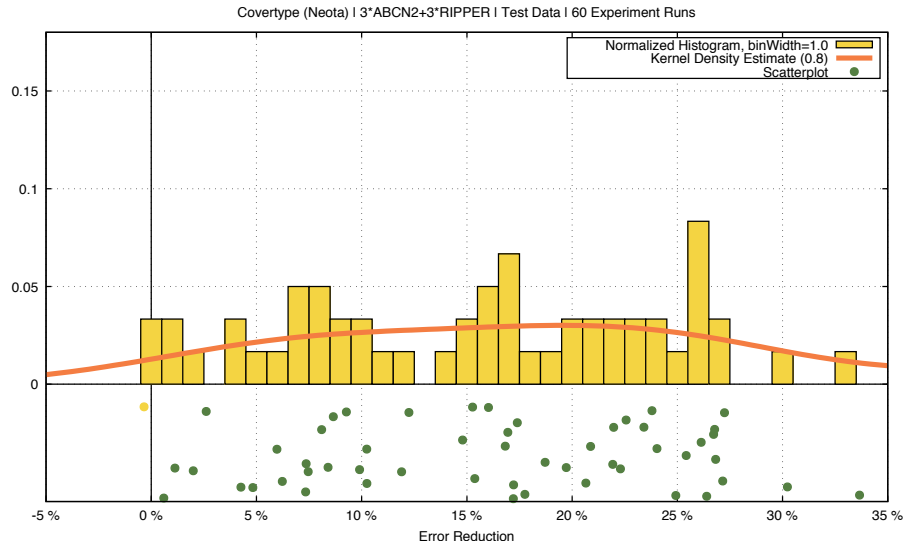
| advisor panel | performance indicator | mean [%] | stddev [%] | min [%] | q25 [%] | median [%] | q75 [%] | max [%] |
|---|---|---|---|---|---|---|---|---|
| Data set: benchmark, Experimental basis: 60 simulation runs | | | | | | | | |
| | *initial accuracy* | 52.034 | 6.863 | 38.65 | 47.123 | 52.135 | 56.489 | 65.589 |
| Group 1: heterogeneous composition of advisor induction algorithms | | | | | | | | |
| 2xJ48+2xABCN2+2xRIPPER | *final accuracy* | 55.056 | 6.928 | 38.951 | 49.866 | 55.807 | 59.511 | 70.527 |
| | *delta accuracy* | 3.022 | 4.138 | -4.757 | 0.758 | 1.818 | 4.452 | 21.759 |
| | *error reduction* | 6.099 | 7.711 | -9.172 | 1.63 | 4.005 | 9.59 | 35.468 |
| 3xABCN2+3xRIPPER | *final accuracy* | 55.037 | 7.491 | 38.745 | 49.102 | 55.939 | 59.511 | 69.626 |
| | *delta accuracy* | 3.002 | 2.798 | -1.656 | 0.826 | 2.683 | 4.268 | 12.491 |
| | *error reduction* | 6.418 | 5.785 | -3.17 | 1.493 | 5.507 | 9.557 | 23.972 |
| 3xJ48+3xRIPPER | *final accuracy* | 53.781 | 6.856 | 38.576 | 48.6 | 54.486 | 58.399 | 67.829 |
| | *delta accuracy* | 1.746 | 3.252 | -4.917 | -0.074 | 1.314 | 2.705 | 12.449 |
| | *error reduction* | 3.466 | 6.558 | -10.992 | -0.121 | 2.572 | 5.314 | 23.892 |
| 3xJ48+3xABCN2 | *final accuracy* | 54.731 | 7.365 | 38.75 | 48.912 | 54.166 | 59.127 | 70.008 |
| | *delta accuracy* | 2.697 | 3.189 | -5.729 | 0.564 | 1.89 | 4.268 | 12.856 |
| | *error reduction* | 5.698 | 6.695 | -13.234 | 1.251 | 3.754 | 9.311 | 24.672 |
| Group 2: homogeneous composition advisor induction algorithms | | | | | | | | |
| 6xABCN2 | *final accuracy* | 54.978 | 6.543 | 41.92 | 49.732 | 55.716 | 59.111 | 66.598 |
| | *delta accuracy* | 2.943 | 3.603 | -0.872 | 0.454 | 2.052 | 3.619 | 19.97 |
| | *error reduction* | 5.898 | 6.547 | -1.889 | 0.904 | 4.108 | 8.188 | 32.55 |
| 6xJ48 | *final accuracy* | 53.334 | 7.114 | 38.389 | 47.338 | 54.428 | 57.01 | 67.796 |
| | *delta accuracy* | 1.299 | 2.988 | -3.095 | -0.315 | 0.325 | 2.174 | 15.847 |
| | *error reduction* | 2.643 | 6.011 | -7.352 | -0.607 | 0.724 | 4.464 | 30.412 |
| 6xRIPPER | *final accuracy* | 53.731 | 7.949 | 38.384 | 47.218 | 53.392 | 59.331 | 70.878 |
| | *delta accuracy* | 1.697 | 2.215 | -2.076 | 0.1 | 0.576 | 3.072 | 7.745 |
| | *error reduction* | 3.848 | 5.038 | -3.759 | 0.183 | 1.343 | 6.892 | 16.74 |

**(b)** Results measured based on a global benchmark dataset.

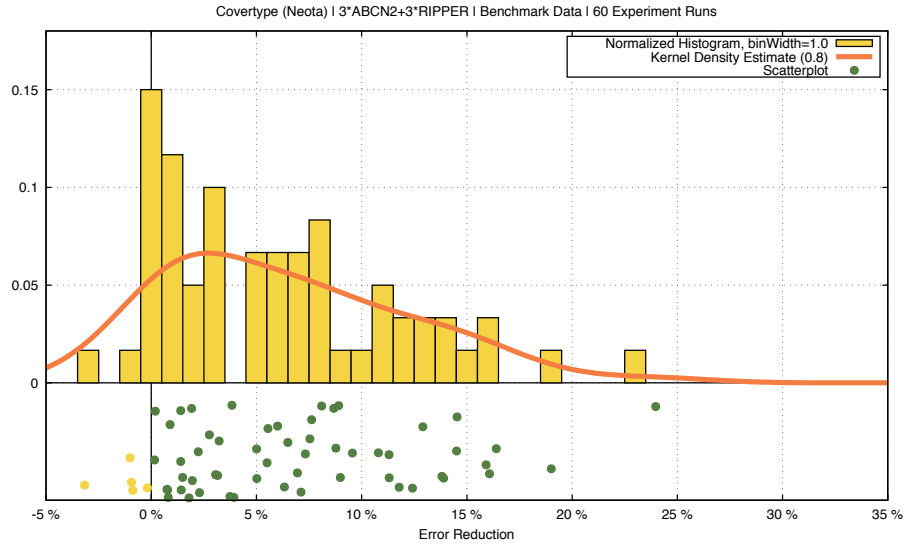
**Table 7.5:** Performance measurements for the Neota land covertype experiment series

another piece of experimental evidence that advisor heterogeneity is a hospitable factor for multiagent interactive adaptation.

Second, Table 7.5a shows that the `2xJ48+2xABCN2+2xRIPPER` and `3xABCN2+3xRIPPER` experiments exceed results of the `6xABCN2` experiment with respect to mean, standard deviation, median, minimum and several quantile values. In terms of mean average delta in classification accuracy, the `3xABCN2+3xRIPPER`




**(a)** Results measured on the basis of the respective advisee test data set limited in scope.




**(b)** Results measured on the basis of a large-scale representative benchmark data set drawn independently from the complete universe of samples.

**Figure 7.29:** Alternative representation of the error reduction measurements for one of the advisee configurations shown in the overview plots discussed earlier. The advisor configuration which has been chosen for closer inspection features two advisors using the J48 algorithm, two using RIPPER and another two using the ABCN2 algorithm. Each data point represents a distinct experiment run.

experiment has the best result with $\mu_{\Delta(acc)} = 8.88 \pm 5.6\,\%$ (cf. Table 7.5a).

Table 7.5 and Figure 7.28 show that the competitive edge of the `2xJ48+2xABCN2+2xRIPPER` and `3xABCN2+3xRIPPER` experiments is retained when the global benchmark dataset is considered as point of reference. Specifically, the latter experiment configuration succeeds to retain a significant delta in average classification accuracy in simulated operationalisation with $\mu_{\Delta(acc)} = 3.0 \pm 2.79\,\%$ and

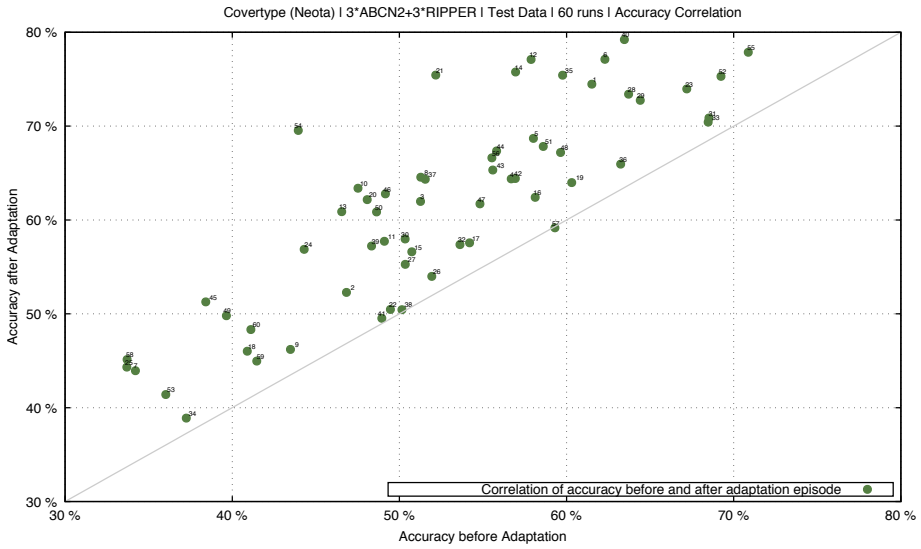



**Figure 7.30:** Correlation of classification accuracy on the advisee test set using the agent's original model (x-axis) and after model refinement (y-axis). The distinct data points have been labeled with their respective experiment run.

$\langle\, q_{10}, q_{90} \,\rangle = \langle\, 0.08, 6.3 \,\rangle\,\%$. Figure 7.28 also suggests that the desired characteristic to mitigate classifier degradation relative to the original base classifier has been procured from the ABCN2-based advisors involved in the experiment.

**Finding**

The takeaway message from the observation of experiments with heterogeneous advisor panels is that the initial positive finding for the FARS 2011 Motorist dataset (cf. Section 7.4.2) has been experimentally confirmed in a second independent task domain. More specifically, the hypothesis that the combination of classification techniques in heterogeneous advisor panels, which is explicitly furthered in the interactive multiagent adaptation framework, is effective, has been proven twice.

### Temporal Correlation of Average Classification Accuracy

Thus far, preceding aspects of the evaluation treated the application of interactive multiagent adaptation with a focus on a bird's eye view on conducted experiments. Now, the same experiments are evaluated from another angle. It is a first tap into the temporal dimension of each experiment run and hence seeks to elucidate the model refinement effects over the course of complete adaptation episodes with their discrete steps (cf. Section 5.3 on page 102).

Figure 7.30 presents a correlation of the average classification accuracy of the original advisee agent model when engaging into a model refinement episode and the adapted version operationalised thereafter. The data in Figure 7.30 is taken from the `3xABCN2+3xRIPPER` experiment which has been identified as the leading heterogeneous setup. The distribution of data points in the correlation plot is

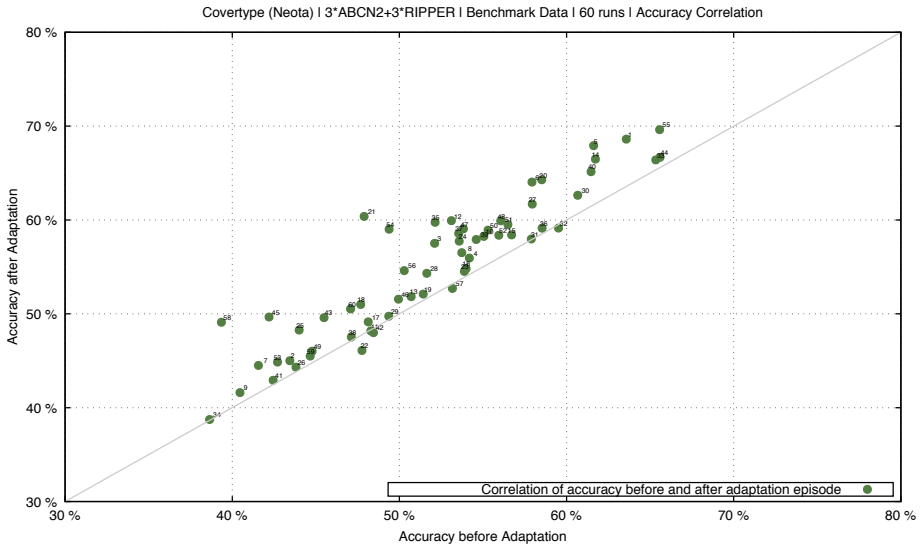



**Figure 7.31:** Correlation of classification accuracy on the global benchmark set using the advisee's original model (x-axis) and after model refinement (y-axis).

also paradigmatic with respect to the other experiments. Hence, this experiment will be discussed further in the following paragraphs.

To begin with, the rough shape of the data point cloud in Figure 7.30 shows several points of differentiation from the analogous data that has been discussed for the FARS 2011 Motorist experiment series. The first difference involves the axis scale. In the Neota land covertype experiments, the original average classification accuracy before any adaptation activities fills a range between $\langle\, 33.7; 70.86 \,\rangle\,\%$ compared to $\langle\, 47.0; 65.47 \,\rangle\,\%$ for FARS (cf. Table 7.2a on page 164). This wide value range conforms to the behaviour of the ABCN2 algorithm that has been observed during the pre-study and is shown in Figure 7.26. The plot based on the advisee test dataset as frame of reference shows that the initial accuracy is improved significantly for many experiment runs across the covered x-axis range. On the lower end of the scale, the adaptation process raises the weakest experiment runs with starting values well below 40 % roughly to or beyond 40 %. In the upper third of the original range, a significant number of experiment runs is pushed into the $\langle\, 70; 80 \,\rangle\,\%$ average accuracy region. Irrespective of the concrete point of origin for an experiment run shown in the plot, the scope of the effect induced by the investigated adaptation episode varies up to about 25 % which is in line with the FARS results.

Another expectation that is confirmed on a limited scale in the plot in Figure 7.30 is that the better the performance of the original model, the smaller the refinement gains. This is evident in the upper right quadrant of the plot where the data points adapt ever more tightly to the diagonal.

A comparison of the plots in Figures 7.31 and 7.30 shows that the overall characteristics of the classifier refinement on the Neota land covertype dataset are quite different when measured based on the global benchmark data set. The data point cloud is shrinked on both axes. This confirms the analysis from the FARS 2011 Motorist experiment series that the comparatively small size of the advisee

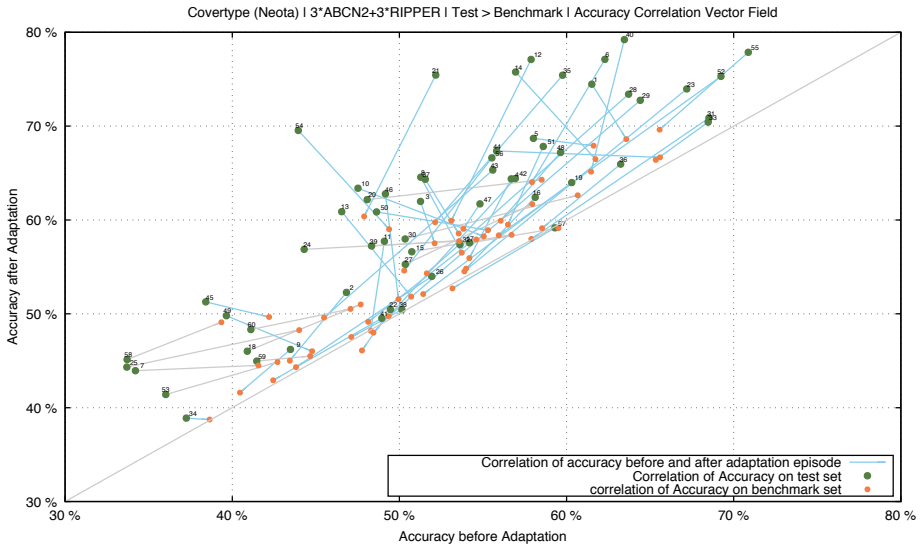



**Figure 7.32:** A complex correlation plot which presents three interrelated analyses within a single graph. First, it correlates the advisee classification accuracy before and after a round of multi-agent interactive adaptation for sixty independent experiment runs as measured based on the respective advisee test data set (green data points). Second, the same correlation has been applied with the larger benchmark data set as basis of measurement (red data points). Finally, data points which belong to the same experiment run have been connected with colour-coded lines. A grey line indicates that the test dataset led to an underestimation of true classification performance, a blue line indicates the opposite case.

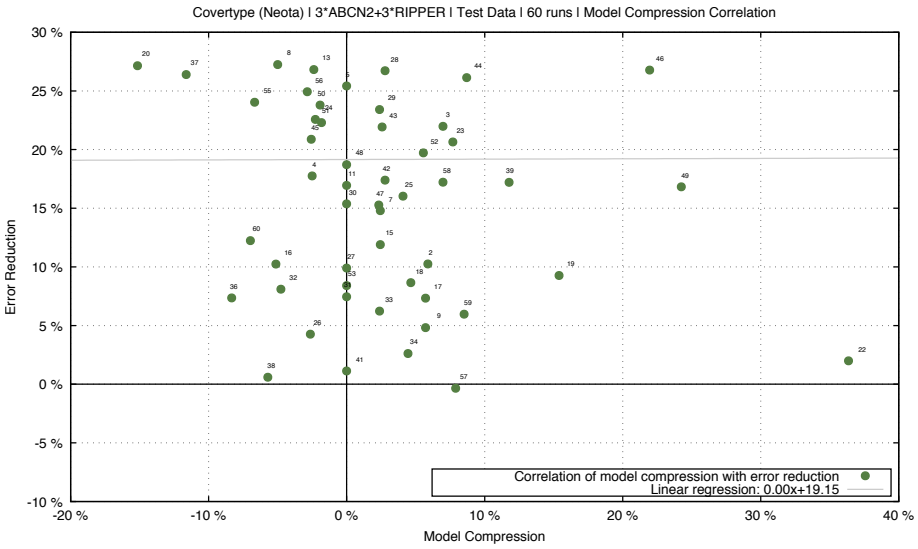
test set is a suboptimal sample draw in order to approximate the characteristics of the agent environment.

The additional plot in Figure 7.32 offers a visual representation of this situation by superimposing the data points for both the experiment run-specific agent test sets and the considerably larger benchmark data set and connecting both data points for the same experiment run.

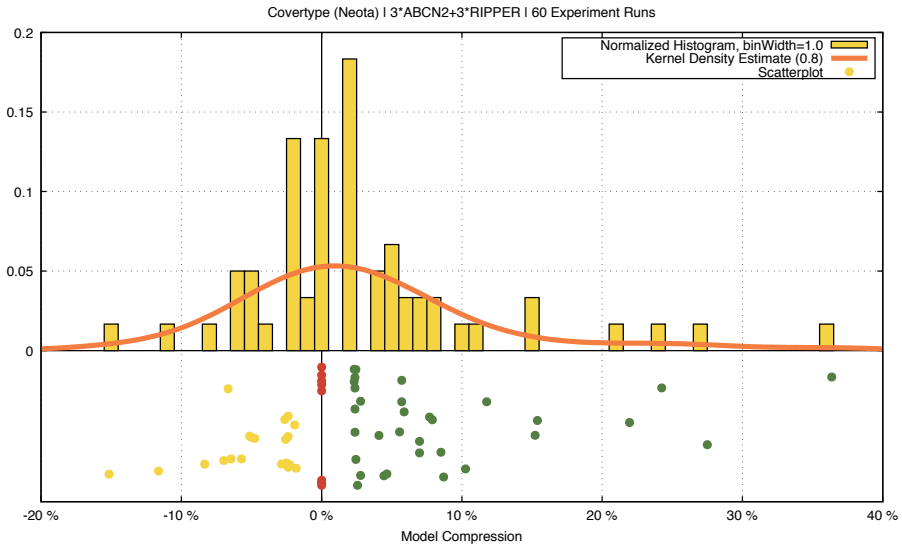
As a result, it is shown that in experiment run such as #25, #24, #20, but also #60, or #15, the measurement for the advisee test set significantly underestimated the actual accuracy of the original advisee model by up to 10 %.

At the other end of the spectrum, experiment runs such as #1, #2 or #35 exhibit the inverse problem with an overestimation of actual accuracy. Without pointing out specific further instances the plot also reveals that the same characteristic error pattern also applies for the estimation of classification accuracy after model refinement.

This observation is significant as it echoes the analysis in Section 7.4.2 which led to the identification of the benefits of a community-based reference dataset for use by advisee agents while engaged in an adaptation episode. The better judgement of re-learned would lead to a focus of the online search process that constitutes the control loop for adaptation episodes (cf. Chapter 5.3).




**(a)** Results measured on the basis of the respective advisee test data set limited in scope. The latter explains the larger deviations from the linear regression function ($f(x) = 0.0 \cdot x + 19.15$) as compared to those for the benchmark set below.




**(b)** Histogram of percentaged rule set compression for a complete experiment series.

**Figure 7.33:** Correlation of the deltas in classification accuracy and model complexity in terms of the number of contributing rules that have been effectuated by multi-agent interactive adaptation. For the FARS 2011 Motorist data set that forms the basis for these experiments, the plot confirms a linear relation among these two key performance indicators. This conclusion is also reflected by both the linear regression and the curve of the weighted cubic splines approximation.

### Correlation of Performance Improvements and Rule Set Compression

For the FARS 2011 Motorist experiment series, the macro-evaluation in Section 7.4.2 found a weak positive correlation between the model compression and the error reduction key performance indicators that was significant specifically for

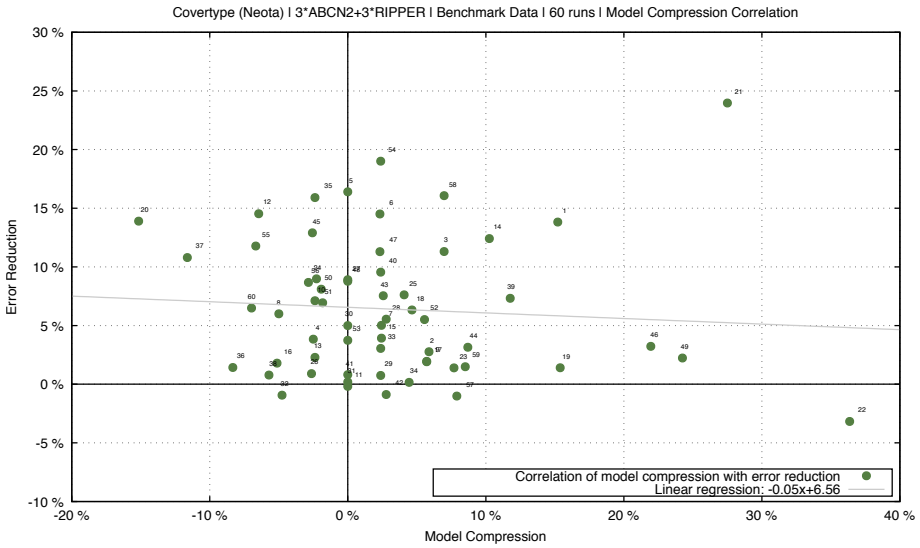



**Figure 7.34:** Result measured on the basis of a large-scale representative benchmark data set drawn independently from the complete universe of samples. Notice that the linear regression function $f(x) = -0.05 \cdot x + 6.56$ sheds a more conservative estimate of the MIA effect in real-world situations.

the advisee test dataset frame of reference. A comparison of Figure 7.33 with the equivalent plots for the Fars experiment series presented in Figure 7.12 reveals that the advisee rule set is affected differently in the respective experiments. Figure 7.33b shows a normalised histogram for the model compression performance indicator measurements for the paradigmatic `3xABCN2+3xRIPPER` experiment. The composite gaussian function obtained by a kernel density estimation has a mean about 0 %. It is nearly symmetric with a prolonged tail in the positive co-domain due to several outlier experiment runs. The number of experiment runs for which the investigated adaptation episode effectuated a deflation or inflation of the original classifier rule set has equal proportions. The described situation deviates notably from the situation that has been observed for the Fars 2011 Motorist experiment series, paradigmatically shown in Figure 7.12b for the `3xJ48+3xABCN2` experiment. There, the distribution centred around a mean at about 8 % model compression. More importantly, the negative co-domain was only populated by a single outlier data point. As a result, rule set deflation has been recorded as the norm, including more significant deflation than in the Neota land covertype, currently under investigation.

A joint inspection of both Figure 7.33b and the plot in Figure 7.33a with this premise shows that the in contrast to the Fars experiment series, even with the advisee test dataset as frame of reference, no meaningful correlation of classification error reduction and rule set deflation/inflation has been recorded in the conducted experiment. This means that notwithstanding the character of quantitative classifier rule set manipulation, a considerable error reduction can result from an adaptation episode. The expectation is, that this effect will surface in the micro-evaluation in Section 7.5.3.

Figure 7.34 constitutes the complimentary plot to Figure 7.33a which is based on

the benchmark frame of reference. While retaining the overall characteristics of the earlier plot such as the missing noteworthy correlation between the model compression and error reduction performance indicator measurements, it can be observed that the value distribution on the y-axis of the plot is by more than a factor 2.5 while remaining almost completely located above the 0 % mark. Consequently, Figure 7.34 conforms with expectations without affording further derivation that have not already been draw already.

**Finding**

The observations for the Neota land covertype dataset can be assessed as an interesting complement to the Fars 2011 Motorist dataset equivalents. While the latter task domain has been instrumental in illustrating expected effects of interactive multiagent adaptation in terms of a deflation of rule sets, the land covertype experiment series stresses that effects can be more diverse in other circumstances, i.e., task domains. Irrespective of advisor configurations, a rule set deflation-oriented behaviour has been discerned as an option rather than a necessity of the implemented adaptation framework. With the land covertype domain, an second option, namely an equal parts inflation and deflation-oriented behaviour has been documented. This finding affirms the decision to evaluate the adaptation framework at hand beyond a single domain and raises further research topics such as the identification of determining factors for one of the observed adaptation behaviours in a task domain.

### Analysis of Search-Driven Adaptation Control Implementation

The evaluation thus far discussed characteristics of the adapted classification models created in an adaptation episode. As a complement, the behaviour of the implemented strategy that drives the control scheme of the adaptation process is investigated with reference to the analogous assessment for the Fars experiment series in Section 7.4.2. As the evaluation is to further understanding on actual search cycles, Table 7.6 provides measurements from search-oriented key performance indicators across experiments with different advisor panels.

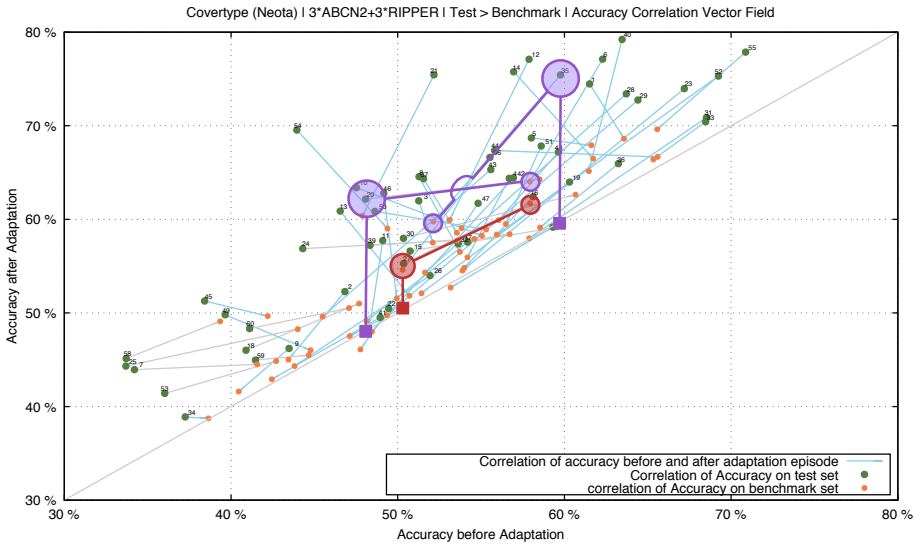
The distribution of search depths in terms of the statistical means listed in the table shows that the search traces in experiment runs are typically rather short with ranging from $\mu_{\text{trace}} = 1.55 \pm 0.6$ and a median of 2 for the `6xRIPPER` experiment to $\mu_{\text{trace}} = 3 \pm 1.35$ and a median of 3 for the `6xJ48` experiment. Hence, for the latter experiment, the length of the search traces has not entailed comparable results in terms of performance-oriented measurements. Rather, `6xJ48` has been among the experiments with the most moderate results in the Neota land covertype experiment series (cf. Section 7.5.2 on page 194). An opposing effect can be observed for the `6xABCN2` experiment. While it surpassed the other two homogeneous experiments in terms of performance (cf. Section 7.5.2 on page 194 ff.), the responsible searches have been comparatively short and focused.

As in the Fars 2011 Motorist experiment series, the involvement of lower-performing induction algorithms for parts of the advisor panel once more, if only slightly, increases the length of search traces in the mean.

| advisor panel | performance indicator | | mean | stddev | min | q25 | median | q75 | max | samples |
|---|---|---|---|---|---|---|---|---|---|---|
| Experimental basis: 60 simulation runs / data | | | | | | | | | | |
| Group 1: heterogeneous composition of advisor induction algorithms | | | | | | | | | | |
| 2xJ48+2xABCN2+2xRIPPER | *search depth* | # | 2.733 | 1.087 | 1 | 2 | 3 | 3 | 6 | |
| | *improving branching* | | | | | | | | | |
| | step #1 \|+ | | 31.77% | 13.91% | 7.02% | 20.97% | 31.34% | 41.67% | 63.08% | 60 |
| | step #2 \|++ | | 15.03% | 10.16% | 1.49% | 6.56% | 13.11% | 20.97% | 40.68% | 57 |
| | step #3 \|+++ | | 6.76% | 5.75% | 1.54% | 2.08% | 4.92% | 9.59% | 24.53% | 31 |
| | step #4 \|++++ | | 4.68% | 2.92% | 1.45% | 1.82% | 3.77% | 5.17% | 9.86% | 8 |
| | step #5 \|+++++ | | 5.29% | 7.52% | 1.79% | 1.85% | 1.92% | 3.39% | 20.59% | 6 |
| | step #6 \|++++++ | | 3.10% | 1.86% | 1.79% | 1.79% | 1.79% | 4.41% | 4.41% | 2 |
| 3xABCN2+3xRIPPER | *search depth* | # | 2.783 | 1.151 | 1 | 2 | 3 | 3 | 7 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 30.07% | 13.28% | 4.29% | 20.37% | 28.13% | 40.00% | 66.67% | 60 |
| | step #2 \|++ | | 15.49% | 10.28% | 1.52% | 9.62% | 13.85% | 20.37% | 48.15% | 57 |
| | step #3 \|+++ | | 6.97% | 5.68% | 1.45% | 3.57% | 4.17% | 8.00% | 22.81% | 32 |
| | step #4 \|++++ | | 4.63% | 4.07% | 1.59% | 1.69% | 2.27% | 4.62% | 12.96% | 8 |
| | step #5 \|+++++ | | 11.58% | 8.06% | 1.75% | 5.88% | 10.17% | 12.73% | 28.81% | 8 |
| | step #6 \|++++++ | | 3.13% | | 3.13% | 3.13% | 3.13% | 3.13% | 3.13% | 1 |
| | step #7 \|+++++++ | | 1.64% | | 1.64% | 1.64% | 1.64% | 1.64% | 1.64% | 1 |
| 3xJ48+3xRIPPER | *search depth* | # | 2.85 | 1.205 | 1 | 2 | 3 | 3 | 6 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 27.43% | 13.97% | 4.00% | 16.67% | 24.14% | 35.29% | 60.61% | 60 |
| | step #2 \|++ | | 13.53% | 10.57% | 1.72% | 5.00% | 10.91% | 18.52% | 44.44% | 54 |
| | step #3 \|+++ | | 6.06% | 6.50% | 1.49% | 1.75% | 3.57% | 8.00% | 30.19% | 34 |
| | step #4 \|++++ | | 5.71% | 3.84% | 1.54% | 2.08% | 4.88% | 8.93% | 14.81% | 15 |
| | step #5 \|+++++ | | 2.53% | 1.28% | 1.47% | 1.47% | 1.92% | 3.77% | 4.48% | 6 |
| | step #6 \|++++++ | | 15.61% | 1.50% | 14.55% | 14.55% | 14.55% | 16.67% | 16.67% | 2 |
| 3xJ48+3xABCN2 | *search depth* | # | 2.633 | 0.901 | 1 | 2 | 2 | 3 | 5 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 31.88% | 15.10% | 5.71% | 20.63% | 30.00% | 41.67% | 66.67% | 60 |
| | step #2 \|++ | | 13.94% | 9.86% | 1.52% | 6.78% | 11.27% | 17.65% | 41.67% | 57 |
| | step #3 \|+++ | | 5.59% | 3.78% | 1.37% | 2.17% | 4.17% | 8.62% | 14.00% | 30 |
| | step #4 \|++++ | | 6.77% | 4.05% | 1.67% | 1.82% | 6.35% | 9.68% | 12.31% | 8 |
| | step #5 \|+++++ | | 16.34% | 25.35% | 1.69% | 1.69% | 1.72% | 45.61% | 45.61% | 3 |
| Group 2: homogeneous composition of advisor induction algorithms | | | | | | | | | | |
| 6xABCN2 | *search depth* | # | 2.383 | 0.865 | 1 | 2 | 2 | 3 | 5 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 18.15% | 9.18% | 2.94% | 11.11% | 17.19% | 25.42% | 37.70% | 60 |
| | step #2 \|++ | | 12.75% | 6.79% | 2.08% | 7.14% | 12.96% | 16.67% | 30.61% | 50 |
| | step #3 \|+++ | | 6.55% | 4.94% | 2.00% | 3.57% | 4.55% | 8.33% | 18.18% | 29 |
| | step #4 \|++++ | | 3.36% | 2.87% | 1.61% | 1.61% | 1.79% | 6.67% | 6.67% | 3 |
| | step #5 \|+++++ | | 3.51% | | 3.51% | 3.51% | 3.51% | 3.51% | 3.51% | 1 |
| 6xJ48 | *search depth* | # | 3 | 1.353 | 1 | 2 | 3 | 4 | 7 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 28.94% | 15.41% | 2.99% | 16.92% | 26.39% | 37.04% | 63.16% | 60 |
| | step #2 \|++ | | 11.69% | 11.20% | 1.45% | 3.23% | 9.23% | 18.03% | 52.38% | 55 |
| | step #3 \|+++ | | 5.76% | 5.30% | 1.20% | 1.69% | 3.33% | 8.00% | 20.00% | 34 |
| | step #4 \|++++ | | 4.50% | 5.18% | 1.32% | 1.67% | 1.92% | 5.36% | 20.00% | 18 |
| | step #5 \|+++++ | | 4.21% | 6.22% | 1.54% | 1.59% | 1.69% | 3.57% | 20.63% | 9 |
| | step #6 \|++++++ | | 5.54% | 6.38% | 1.72% | 1.72% | 2.00% | 12.90% | 12.90% | 3 |
| | step #7 \|+++++++ | | 1.56% | | 1.56% | 1.56% | 1.56% | 1.56% | 1.56% | 1 |
| 6xRIPPER | *search depth* | # | 1.55 | 0.594 | 1 | 1 | 2 | 2 | 4 | |
| | *improving branching* | % | | | | | | | | |
| | step #1 \|+ | | 18.40% | 10.58% | 1.43% | 8.45% | 18.92% | 26.79% | 44.07% | 60 |
| | step #2 \|++ | | 11.50% | 8.22% | 1.82% | 4.44% | 10.42% | 15.38% | 34.15% | 31 |
| | step #3 \|+++ | | 4.88% | | 4.88% | 4.88% | 4.88% | 4.88% | 4.88% | 1 |

**Table 7.6:** Search-oriented overview of experiment series, providing details for search depths and branching.

This observation is accompanied by measurements for the fraction of improving branching options for the different advisor compositions. Table 7.6 breaks down the fractions for the respective adaptation steps. Compared with the FARS Motorist case, the data shows that the initial fraction of improving branching options with $\mu \in \langle\, 18.15; 28.94 \,\rangle$ percent are much higher, especially for the J48 advisor




**(a)** Experiment `3*ABCN2+3*RIPPER`

**Figure 7.35:** The experiment correlation plot serves to highlight experiment runs that are subject to the following micro evaluation.

panel. However, the decline of further improving branching options in subsequent adaptation steps is stark. An observation that is confirmed by the Neota data is the effect of advisor heterogeneity on the fraction of improving branching options, even pushing the median in three out of four heterogeneous setups beyond a mean of 30%. These numbers suggest, that advice from agents using different induction techniques help to address complementary critical examples presented by the advisee. However, this beneficial initial situation does for the given domain not translate to a more comprehensive overall search state exploration that would alleviate early exploration costs.

**Finding**

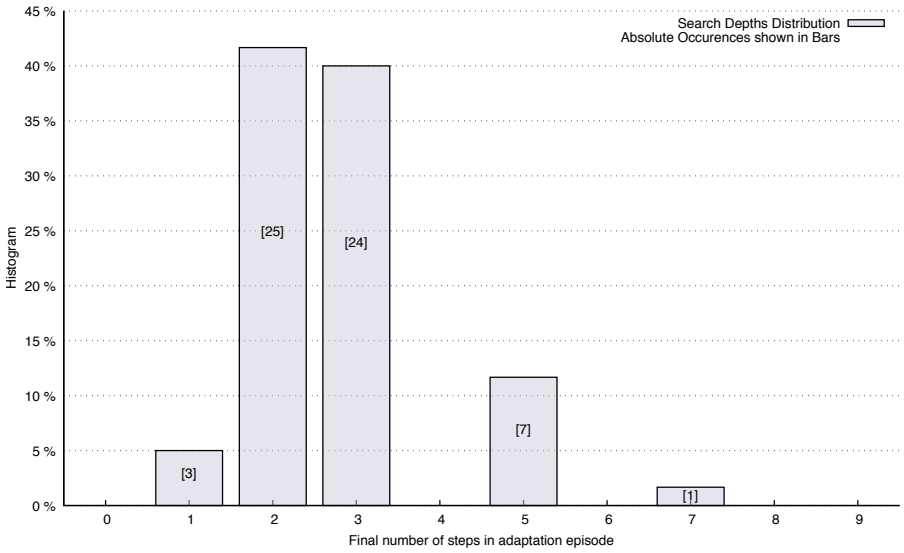
The results above affirm the indications from Section 7.4.2 for a second task domain that algorithm heterogeneity among agents in an advisor panel is a factor promoting a more thorough exploration of the classifier search space.

The experiments have also affirmed that the overall branching factor throughout adaptation episodes is considerable and leads to a costly exploration for each search step.

### 7.5.3 Micro Evaluation of Selected Experiment Runs

As for the Fars 2011 Motorist experiment series, the evaluation proceeds with a micro evaluation to further understanding of the effects of classification model adaptation episodes as they advance in distinct refinement steps.

The evaluation runs that are investigated are highlighted in Figure 7.35. The plot




**(a)** Experiment `3*ABCN2+3*RIPPER`

**Figure 7.36:** Histogram of search depths.

refers to the advisee test data set as reference frame, i.e., the micro evaluation adopts the perspective of the agent seeking to enhance its classifier. The discussion includes two successful experiment runs from the agent perspective (i.e., run #20 and #35 from `3*ABCN2+3*RIPPER`). Also, it includes a paradigmatic run that can be classified as only marginally successful (run #27 from `3*ABCN2+3*RIPPER`).

Before delving into the micro evaluation entries, the visual juxtaposition of the correlation plots from two experiments in the same experiment series in Figure 7.35 illustrates one effect of the determinism in control the configuration of seeded random processes for data sampling across experiments, which has been specifically developed for this evaluation, as explained in Section 7.3. A comparison of the x-coordinates of data points in the plots reveals conformance throughout. The prerequisites for experiment runs even across experiments are replicated exactly, thus establishing an experiment fixture and allowing to focus fully on the parameterisation induced by the different advisee panel compositions (cf. Figure 7.27 on page 195).

### Experiment-Level Evaluation · `3*ABCN2+3*RIPPER`, Run #20

A first experiment run chosen for detail evaluation is run #20 from the experiment that used a mixed advisor panel of three ABCN2-enabled advisors and three using theRIPPER algorithm (`3xABCN2+3*RIPPER`, cf. Figure 7.36a).

**Inspection of Model Evolution** Figure 7.37 illustrates that the investigated model refinement episode is composed of a series of seven discrete refinement steps. The number of steps thereby corresponds to the depth of the search trace in the local search strategy (cf. Section 5.3). As can be read off the search depth histogram presented in Figure 7.36a, run #20 is a paradigmatic example for an extraordinarily long search trace.

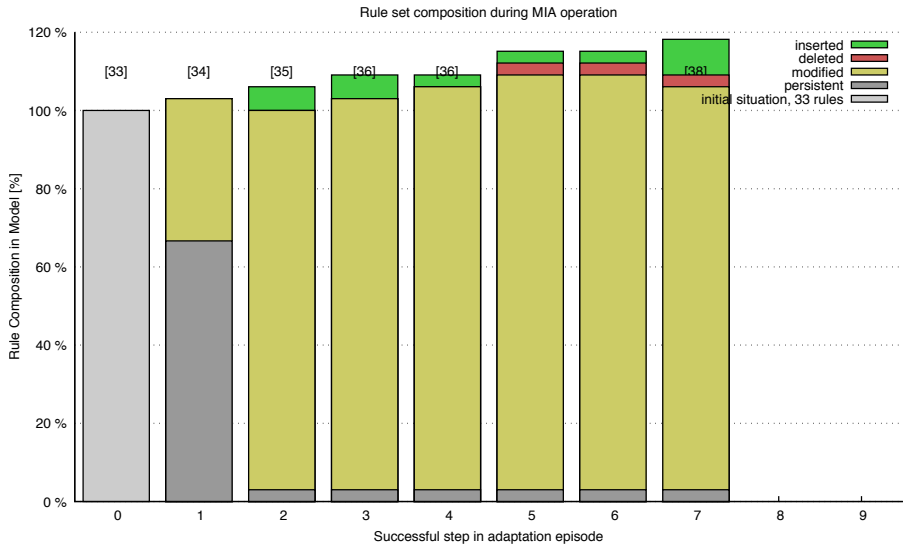
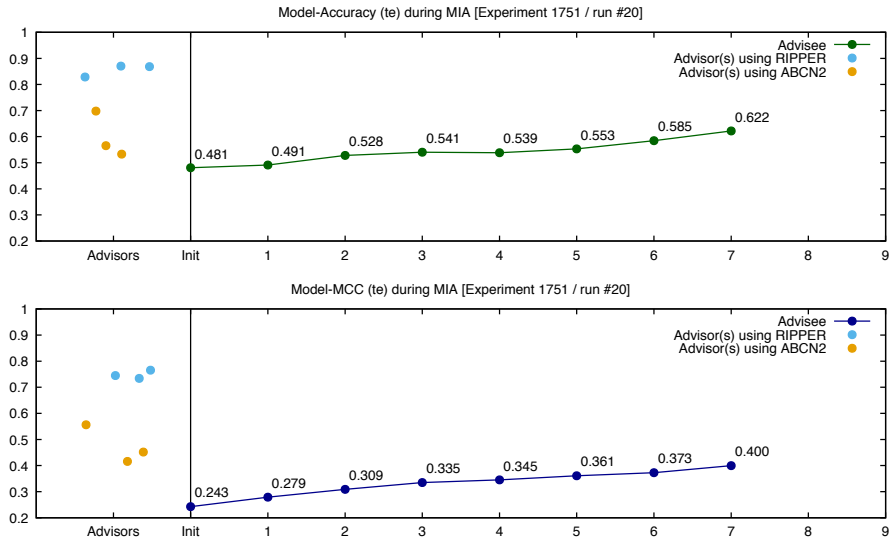



**Figure 7.37:** A plot with a detail analysis for a single experiment run from the experiment series `3xABCN2+3xRIPPER` (run #20). It breaks dow the modification of an advisee model during an adaptation episode. To quantify the change of the advisee rule set, the rules for different interaction steps have been ordered and compared by means of a diff-algorithm, hence coarsely highlighting rule change, rule removal and rule generation.

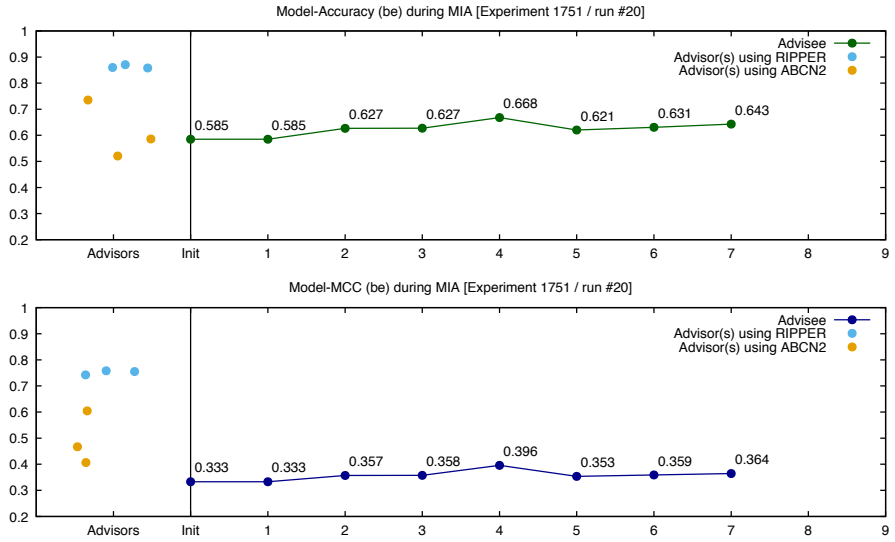
Turning the attention to the model change behaviour, the data suggests that a large part of the model refinement occurs in the first two process steps. It is shown, that the original rule base is modified to a large extent, while the total number of rules rises by two rules. In the remaining adaptation steps, the number of rules rises further to a maximum of 38 at the end of the adaptation episode. It is assumed that in the last steps the rule set modification becomes more focussed. This assessment is consistent with both Figure 7.37 and Figure 7.38 on the next page.

**Evolution of Classifier Performance** The plot of the development of average predictive accuracy over the course of the adaptation episode and its steps in Figure 7.38a supplements the data presented in Figure 7.37 and the analysis presented thus far. The performance increases monotonously across all steps, starting from a very modest initial $mcc = 0.243$ and an average accuracy of 48.1 %. The growth measured in terms of both key performance indicators is roughly linear in all steps with a very shallow inclination.

The comparison of the advisee measurements with those from the agents in the acting advisor panel is essential to set any performance gains discussed so far in context. The data suggests that the naive advisee is only able to raise its perceived predictive accuracy up to the level of the weakest ABCN2-powered member of the advisee panel. Hence, for this experiment run, the adaptation episode, while benefitting the agent, nevertheless fails to propel it to a level that could be described as average among advisors. More specifically, the supposedly superior predictive accuracy of the RIPPER advisors is not approached.




**(a)** Advisee/advisor test data set frame of reference

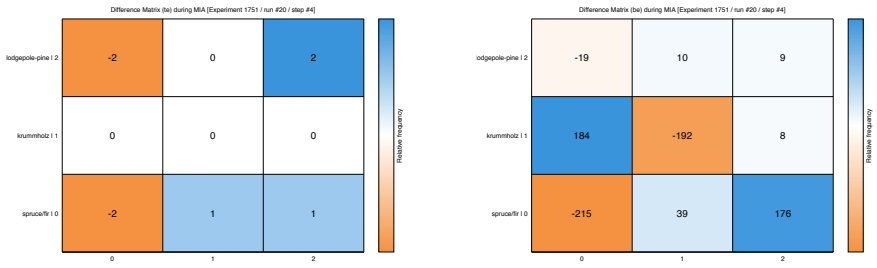



**(b)** Global benchmark data set frame of reference

**Figure 7.38:** Development of the advisee rule induction performance as measured with respect to accuracy (top plot) and Matthews Correlation Coefficient (cf. Section 5.3.1, bottom plot) against the test data set. The latter has been used in the examined experiment setup as the performance indicator that informed search process. The invariant advisor performance measurements are rendered on the left respectively.

It can be assumed that the advisee spends a considerable effort to only minor success by concentrating on quite specific problematic instances.

To complement the critical assessment of the performance evolution throughout the adaptation episode, it is instrumental for the discussed experiment run to oppose the agent test data frame of reference for the classifier performance evolution as expressed in Figure 7.38a with the global benchmark data frame of reference for measurements, shown in Figure 7.38b. This juxtaposition reveals a



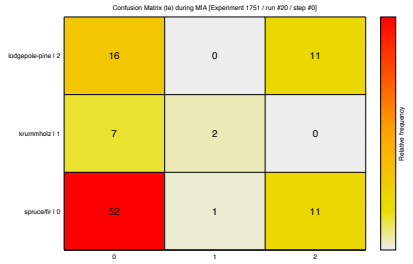

**(a)** Adaptation Step #4 (Test Data). **(b)** Adaptation Step #4 (Benchmark Data).

**Figure 7.39:** Detailed picture of the benchmark regression in step #4 of the adaptation episode.

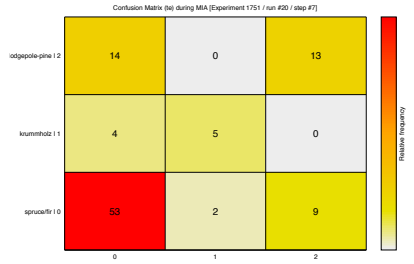
deviation in the performance developments that has not yet been encountered in those detail investigations performed in the context of the respective micro evaluation for the FARS 2011 Motorist data set (cf. Section 7.4.3). The plot shows that the monotonous increase in performance measured on the advisee test set is not replicated against the benchmark data. Instead, adaptation step #5 introduces a notable regression in classifier performance which subsequent steps mitigate but cannot revert fully (cf. Figure 7.39 on page 211). Hence, the data shows that an operationalisation of an intermediate adapted classifier would have yielded slightly higher-quality predictions than the final classifier. The ex-post assessment affirms an earlier criticism of the applied adaptation prerequisites, specifically with the degree to which the advisee test set is representative of the situation when a revised classifier is operationalised. If the approximation is weak, regressions as in run #20 cannot be precluded.

**Result Visualisation by Means of Confusion Matrices** As the detail evaluation of the first experiment run already offered new insights into the classifier refinement process and issues therein, this line of analysis should be carried forward as the focus now turns on the discussion of confusion matrices beginning with those for the advisee test set.

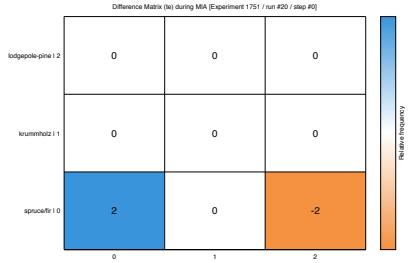
The confusion matrices in Figure 7.40 shows that the adaptation episode by the advisee addresses each of the three target classes no matter their majority or minority status. This is documented specifically in the difference matrices for the adaptation steps #1 to #7. These steps can further be categorised in two classes. First, adaptation steps #1, #2, #3 and #7 effectuate elimination of false-negative errors for the respective target class addressed in a learning problem. Second, the remaining adaptation steps #4 till #6 effectuate changes two two target classes, with one being the intended target while the other is collaterally affected. This state of affairs is shown best in Figure 7.40i. While successfully addressing `lodgepole-pine` mis-classifications in adaptation step #5, an equal amount correct classifications for the `spruce/fir` majority class regresses to false-negative classifications. These imperfect adaptation steps partly cancel each other out and finally result in an impure cumulative difference matrix in Figure 7.40j.



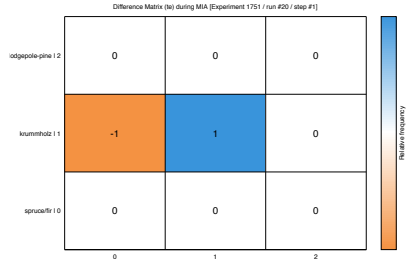

**(a)** Initial situation.



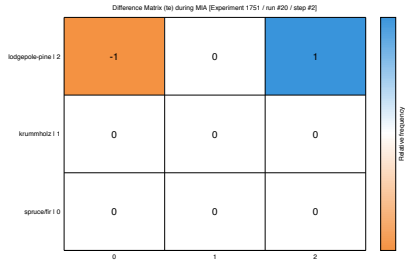

**(b)** Final situation.



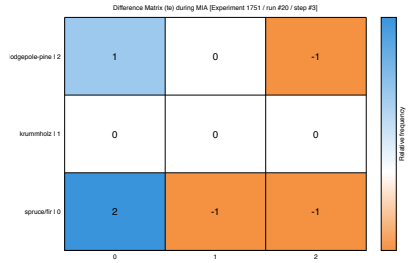

**(c)** Adaptation Step #1.



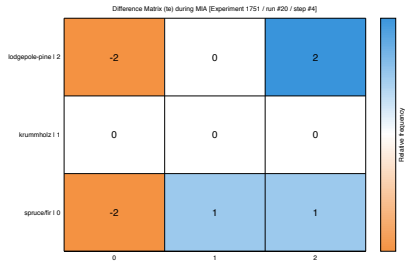

**(d)** Adaptation Step #2.



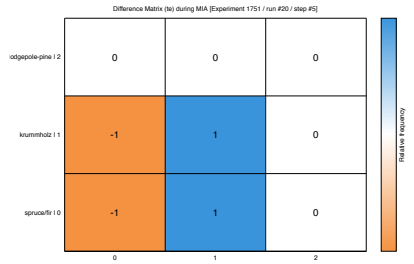

**(e)** Adaptation Step #3.



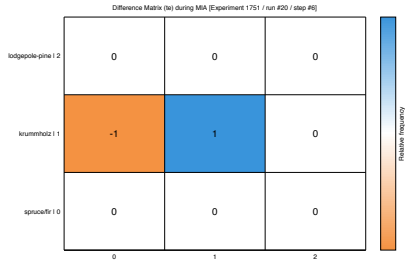

**(f)** Adaptation Step #4.




**(g)** Adaptation Step #5.




**(h)** Adaptation Step #6.




**(i)** Adaptation Step #7.


**(j)** Cumulative difference matrix for adaptation steps

**Figure 7.40:** Visualisation of the model adaptation process for experiment run #20 in the `3xABCN2+3xRIPPER` experiment series.

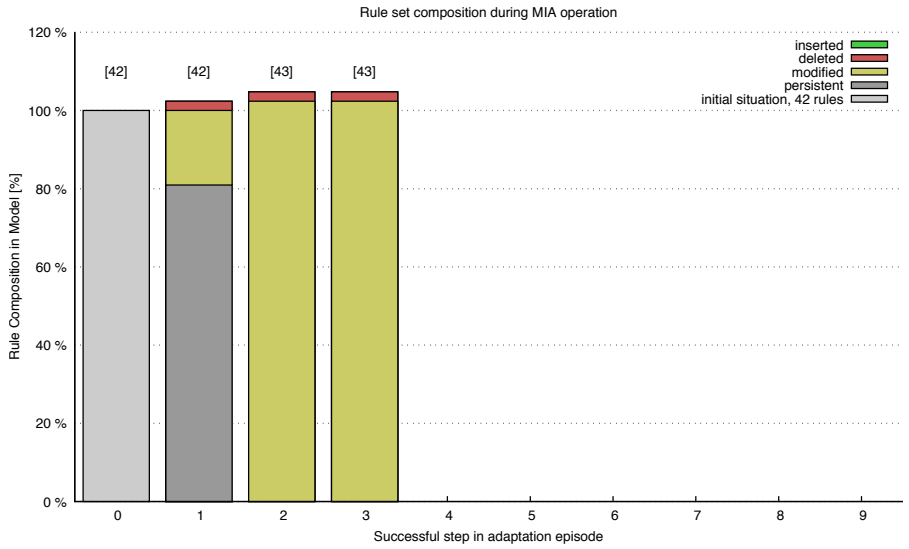



**Figure 7.41:** A plot with a detail analysis for a single experiment run from the experiment series `3xABCN2+3xRIPPER` (run #35). It breaks dow the modification of an advisee model during an adaptation episode. To quantify the change of the advisee rule set, the rules for different interaction steps have been ordered and compared by means of a diff-algorithm, hence coarsely highlighting rule change, rule removal and rule generation.

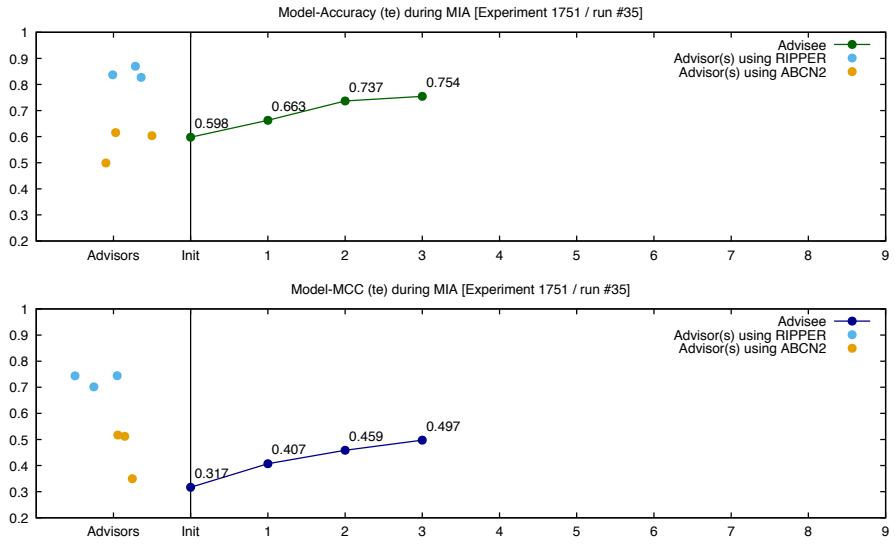
### Experiment-Level Evaluation · `3*ABCN2+3*RIPPER`, Run #35

The second experiment run chosen for detail evaluation is run #35 from the experiment that used a mixed advisor panel of three ABCN2-enabled advisors and three using RIPPER algorithm (`3xABCN2+3*RIPPER`, cf. Figure 7.36a).

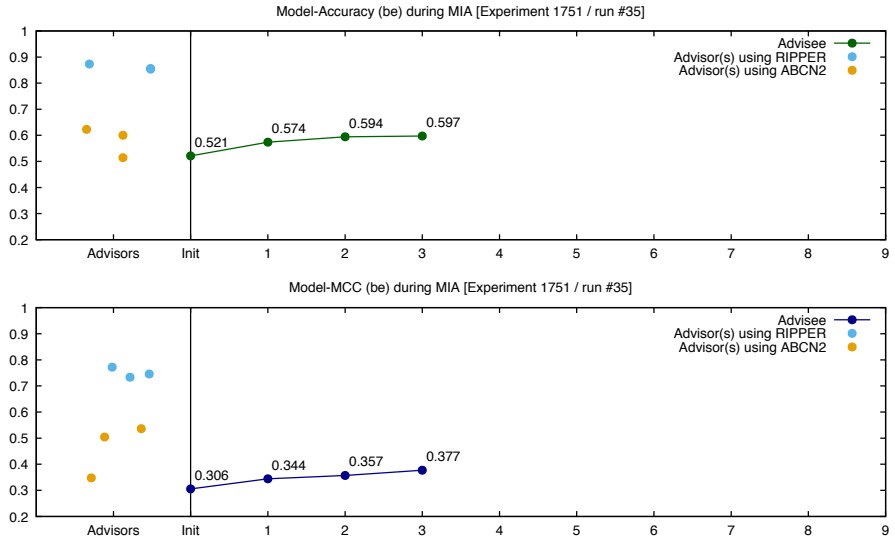
**Inspection of Model Evolution** Figure 7.41 illustrates that the investigated model refinement episode is composed of a series of three discrete refinement steps. As can be read off the search depth histogram presented in Figure 7.36a, run #35 is a paradigmatic example for an average length search trace.

With regard to the model change behaviour, the data from the experiment runs shows that the change induced by the first adaptation step involves about 20 % of the original rule base which is a lesser number compared to the previously examined experiment run. With the following second adaptation step, however, the rule base is rearranged completely without either a significant change in the total number of rules which only grows by a single rule. In the final step, the change process supposedly continues. In total, this would lead to the expectation of three back to back adaptation steps with consistent performance gains. This assessment is consistent with both Figure 7.41 and Figure 7.42 on the following page.

**Evolution of Classifier Performance** The plot of the development of average predictive accuracy over the course of the adaptation episode and its steps in Figure 7.42a supplements the data presented in Figure 7.41 and the analysis presented thus far. The performance increases monotonously across all steps,




**(a)** Advisee/Advisor test data set frame of reference




**(b)** Global benchmark data set frame of reference

**Figure 7.42:** Development of the advisee rule induction performance as measured with respect to accuracy (top plot) and Matthews Correlation Coefficient (cf. Section 5.3.1, bottom plot) against the test data set. The latter has been used in the examined experiment setup as the performance indicator that informed search process. The invariant advisor performance measurements are rendered on the left respectively.

starting from a $mcc = 0.317$ and an average accuracy of 59.8 %. The growth measured in terms of both key performance indicators is notable in all steps with a slight flattening for the last adaptation step.

The experiment run confirms the observation from the first investigated run that the performance improvement of the advisee leads to an approximation of the performance of the best ABCN2 advisor which belongs to the less capable induction algorithm group in the advisor panel. That is, especially then concentrating on the

Matthews Correlation Coefficient metric, the performance of the RIPPER advisors cannot be reached. What is interesting though is the fact that for experiment run #35, the measurements for the global benchmark data set confirm the trend measured based on the advisee test data set. Hence, run #35 constitutes an example for the desirable effect that the operationalisation of a modified classifier, even without further future adaptation yields a strategic benefit for the advisee.

Indeed, the direct juxtaposition of results from run #20 and #35 again sheds light on the uncertainty in the operationalisation of a modified advisee model. This illustrates the demand for future work to improve the return on invest in operationalisation.

**Result Visualisation by Means of Confusion Matrices** As the detail evaluation of the first experiment run already offered new insights into the classifier refinement process and issues therein, this line of analysis is carried forward as the focus now turns on the discussion of confusion matrices beginning with those for the advisee test set.

The confusion matrices in Figure 7.43 show that the adaptation episode by the advisee addresses each of the three target classes no matter their majority or minority status. This is documented specifically in the difference matrices for the adaptation steps #1 and #3. Unlike in the previously investigated experiment run, the elimination of false classifications is not accompanied with regressions. Consequently, the adaptation episode is well focussed on addressing specific learning problems, in the process enhancing class boundaries such that additional instances in other classes get correctly classified as well.

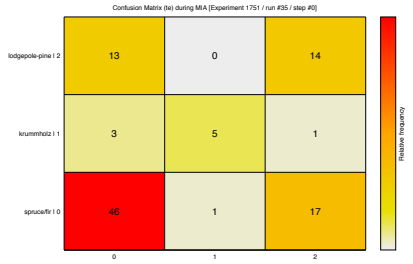
### Experiment-Level Evaluation · `3*ABCN2+3*RIPPER`, Run #27

The third experiment run selected for a detail evaluation is run #27. It also belongs to the `3xABCN2+3xRIPPER` experiment series. However, as Figure 7.35 highlights, this is an experiment run where the positive effect of the classifier adaptation episode is not as pronounced as in the experiment runs discussed this far.

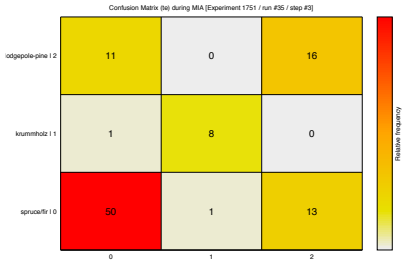
**Inspection of Model Evolution** Figure 7.44 illustrates that the investigated model refinement episode is composed of a series of only two discrete refinement steps. According to the search depth histogram presented in Figure 7.36a, run #27 is an example for the most frequent search trace length with 25 out of 60 experiment runs.

With regard to the model change behaviour, the data from the experiment runs shows that the pattern of change induced the two adaptation steps is quite similar to that previously discussed for experiment run #27. Again, the initial change involves about 25 % of the original rule base. Then, with the following second adaptation step, the rule base is rearranged to an extent of more than 95 %. In both steps the size of the rule base is kept constant with only a single added rule.

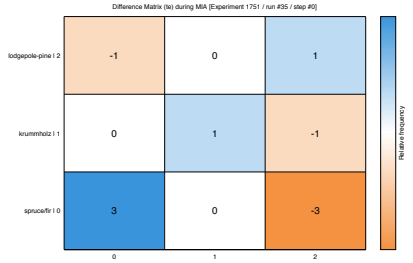
**Evolution of Classifier Performance** The plot of the development of average predictive accuracy over the course of the adaptation episode and its steps in



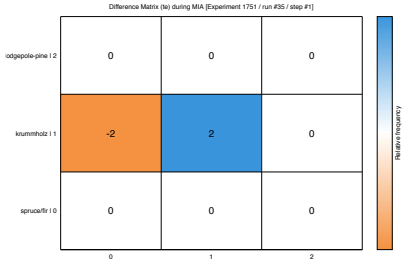

**(a)** Initial situation.



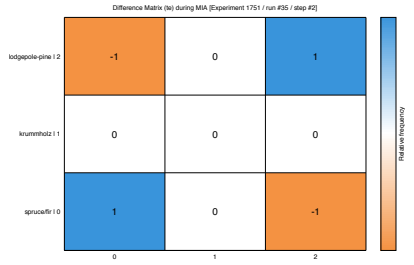

**(b)** Final situation.



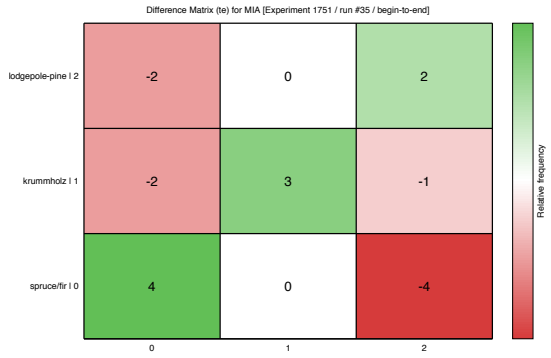

**(c)** Adaptation Step #1.




**(d)** Adaptation Step #2.




**(e)** Adaptation Step #3.




**(f)** Cumulative difference matrix for adaptation steps

**Figure 7.43:** Visualisation of the model adaptation process for experiment #35 in the `3xABCN2+3xRIPPER` experiment series. The first row of matrices directly contrasts the test set result obtained with the initial classifier with the result for the final modified version. The four matrices below constitute difference matrices that show the effects of the respective steps within the adaptation episode. The final matrix consequently accumulates these discrete steps.

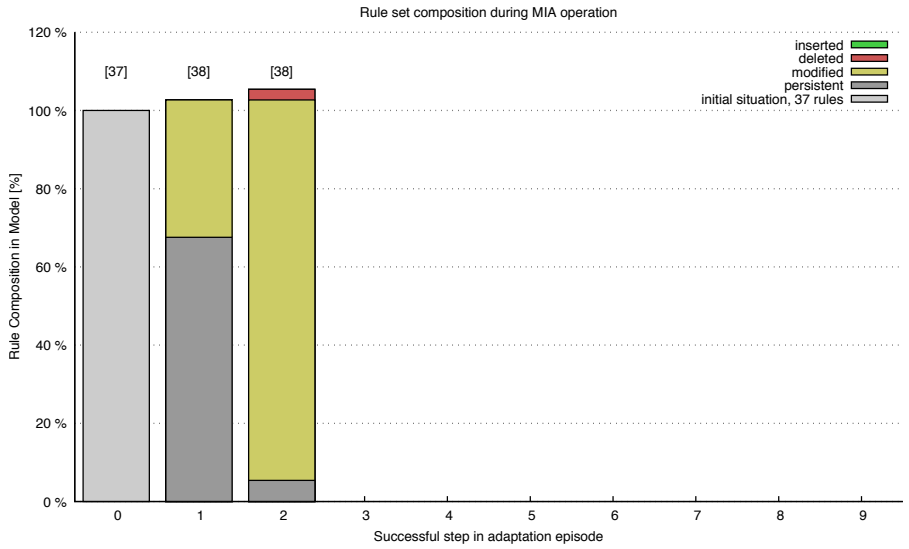



**Figure 7.44:** A plot with a detail analysis for a single experiment run from the experiment series `3xABCN2+3xRIPPER` (run #27). It breaks dow the modification of an advisee model during an adaptation episode. To quantify the change of the advisee rule set, the rules for different interaction steps have been ordered and compared by means of a diff-algorithm, hence coarsely highlighting rule change, rule removal and rule generation.

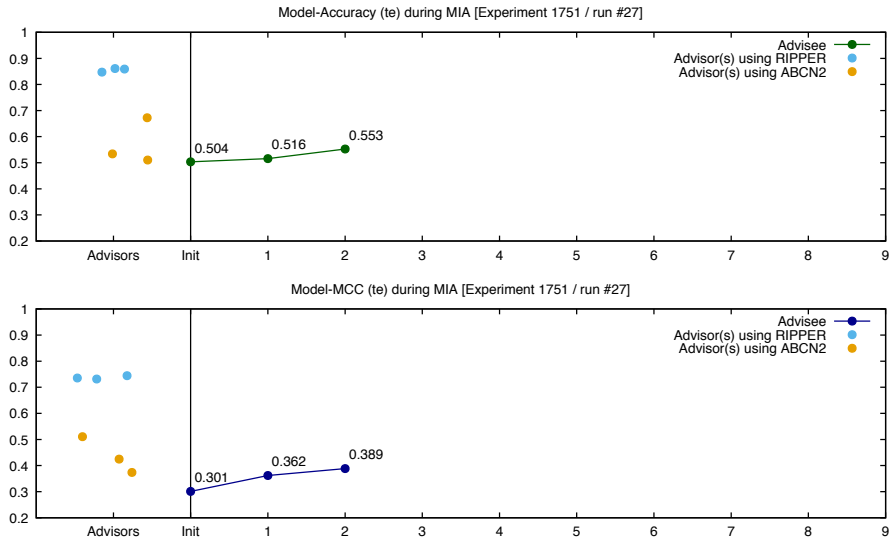
Figure 7.45a supplements the data presented in Figure 7.44 and the analysis presented thus far. The performance increases monotonously in both steps, starting from an $mcc = 0.301$ and an average accuracy of 50.4 %. The growth measured in terms of both key performance indicators is less distinctive than in the previous experiment run.

Once more, the advisee can improve its classifier only in the performance area marked by the ABCN2 advisor, by now a recurrent observation in the micro evaluation for the land covertype dataset.

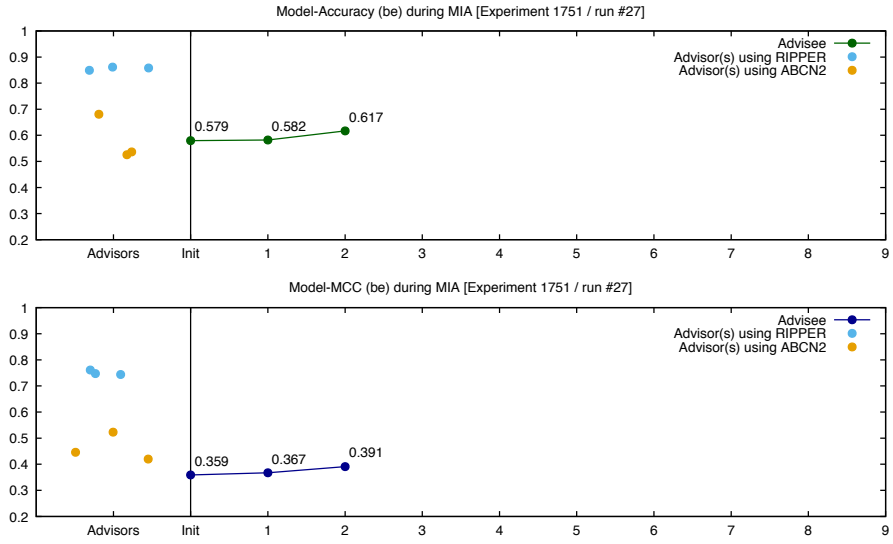
What can be stressed for the experiment run at hand is the fact that the global benchmark set frame of reference confirms the measurements based on the advisee test set. Hence, although rather short, the effect of the adaptation episode translates well into projected operationalisation. This is in spite of the fact, that the measurements against the advisee frame of reference underestimated the true performance of the original classifier.

**Result Visualisation by Means of Confusion Matrices** The confusion matrices in Figure 7.46 stress the modest character of the enhancement induced by the advisee's investigated adaptation episode for the advisee test data frame of reference. Compared to the previous experiment run #35 (cf. Figure 7.46), only a third of the instances is forced to a correct classification.

Both adaptation steps focus on a learning problem for the `spruce/fir` class (run #1) and the `krummholz` class respectively without collaterals. The `lodgepole-pine` class on the other hand is not affected at all.




**(a)** Advisee/Advisor test data set frame of reference

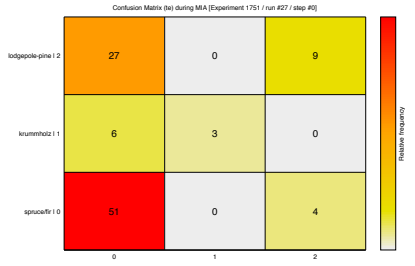



**(b)** Global benchmark data set frame of reference

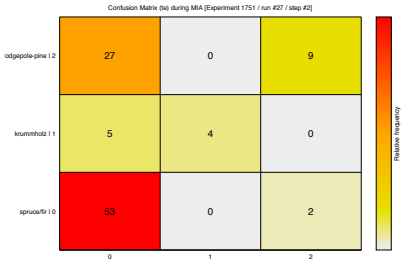
**Figure 7.45:** Development of the advisee rule induction performance as measured with respect to accuracy (top plot) and Matthews Correlation Coefficient (cf. Section 5.3.1, bottom plot) against the test data set. The latter has been used in the examined experiment setup as the performance indicator that informed search process. The invariant advisor performance measurements are rendered on the left respectively.

## 7.5.4 Discussion of Findings

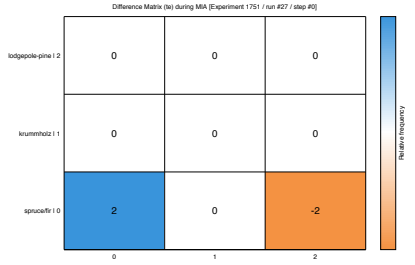
Following the conduct of both stages of the evaluation of the Neota Covertype dataset in the preceding Sections 7.5.2 and 7.5.3, the gist of findings can be presented.



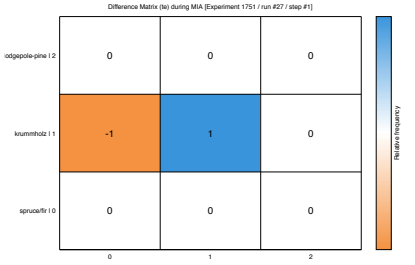

**(a)** Initial situation.



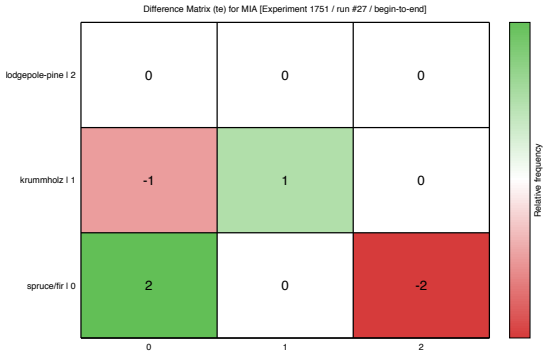

**(b)** Final situation.




**(c)** Adaptation Step #1.




**(d)** Adaptation Step #2.




**(e)** Cumulative difference matrix for adaptation steps

**Figure 7.46:** Visualisation of the model adaptation process for experiment #27 in the `3xABCN2+3xRIPPER` experiment series. The first row of matrices directly contrasts the test set result obtained with the initial classifier with the result for the final modified version. The four matrices below constitute difference matrices that show the effects of the respective steps within the adaptation episode. The final matrix consequently accumulates these discrete steps.

**Findings for the Macro Evaluation** Concluding the macro-evaluation for the Neota dataset, the takeaway message from the observation of experiments with heterogeneous advisor panels is that the initial positive finding for the Fars 2011 Motorist dataset (cf. Section 7.4.2) has been experimentally confirmed in a second independent task domain. More specifically, the hypothesis that the combination of classification techniques in heterogeneous advisor panels is effective, has been proven twice.

The recorded data for the homogeneous Neota land covertype experiments has shown a ranking that is reciprocal to the performance measurements for respective learning agents in the pre-study. The results confirm that RIPPER and J48-based advisors have not been able to procure advice for the presented learning problems whose consideration as background knowledge in re-learning processes led to novel generalisations that boosted classifier performance beyond the level of the ABCN2 experiment.

Consequently, the experiments suggest that the utility of an advisor in the investigated implementation of interactive multiagent adaptation of individual classification models is not only dependent on the standalone performance of the advisor model, but on the process of concrete advice synthesis for given learning problems.

The observations for the Neota land covertype dataset constitute an interesting complement to the FARS 2011 Motorist dataset equivalents. While the latter task domain has been instrumental in illustrating expected effects multiagent interactive adaptation in terms of a deflation of rule sets, the land covertype experiment series stresses that effects can be more diverse in other circumstances, i.e., task domains. Irrespective of advisor configurations, a rule set deflation-oriented behaviour has been discerned as an option rather than a necessity of the implemented adaptation framework. With the land covertype domain, an second option, namely an equal parts inflation and deflation-oriented behaviour has been documented.

This finding affirms the decision to evaluate the adaptation framework at hand beyond a single domain and raises further research topics such as the identification of determining factors for one of the observed adaptation behaviours in a task domain.

**Findings for the Micro Evaluation** Concluding the micro-evaluation for the experiment runs #20, #35, and #27, the differentiated impression reported for the FARS experiment series also arises for the Neota land covertype experiment series. While the positive effects on an underperforming initial advisee classifier have been carved out in the paradigmatically discussed experiment runs, the same runs have also served to illustrate drawbacks of the investigated adaptation approach.

The graphical development of classifier performance in run #20, illustrated in Figure 7.38, has shown by example the challenges with respect to the knowledge based control of the adaptation process. The bounded knowledge of the advisee agent during adaptation leads to an extended multi-step adaptation while the matching stepwise simulation of an adapted model operationalisation suggests that it is exactly the advisee persistence that incurs a notable performance regression for operationalised classifier performance.

Run #27 confirmed an effect already known from the FARS 2011 Motorist data set where the exploration of the model search space is stopped very soon as performance improvements level off almost right from the start.

Hence, even though encouraging runs such as in run #37 have been documented, that improve the initial advisee classifier significantly when measured against the advisee test dataset frame of reference and also translate well simulated

operationalisation measured against the benchmark dataset frame of reference, the assessment conducted for the micro evaluation is a reminder that the robustness of the tested adaptation approach remains an important topic in the future.

## 7.6 Chapter Summary

The concept for interactive multiagent adaptation of individual classification models has been subjected to an extensive empirical evaluation in simulation studies. To address the multidimensional research question on the feasibility of the adaptation methodology proposed in the thesis, important groundwork as enabler for the controlled evaluation of machine learning centred simulation studies needed to be procured. Consequently, Section 7.3 delineated contributions to extend the existing general-purpose multiagent-based simulation system PlaSMA to support modelling, configuration, reproducible conduct of simulation experiments, and their assessment based on key performance indicators. The PlaSMA data base model has been extended such that arbitrary classification task domains can be represented. Large data repositories for these task domains can be uploaded following suitable data cleansing and preprocessing. A dedicated data warehousing agent has then been introduced with the ability of unbiased sampling from a given data repository for a task domain according to specifications as part of the agent configuration. By means of this environment agent which is exposed to regular simulation actors as service provider, an environmental abstraction has been introduced for PlaSMA scenarios which enables the focussed investigation of well-defined situations in dynamic knowledge networks with a number of constituent learning agents. Thus, the point of departure for the empirical evaluation in this chapter has been established.

Based on these groundwork contributions, a large-scale experiment series has been conducted according to the experimental setting in Section 7.2.2 for two independent task domains and real-world data repositories. These task domains comprise the FARS 2011 Motorist dataset (cf. Section 7.4.1) and the Neota land covertype dataset (cf. Section 7.5.1). In total, the conduct of the experiment series for both domains involved 840 simulation runs where each run represents a specific initial situation where an adaptation episode is conducted.

**Assessment from the Knowledge Management Perspective** The evaluation for these experiment series in Sections 7.4 and 7.5 answers the research question posed initially in Section 7.1 from the agent-oriented knowledge management perspective whether the process of knowledge creation can be effectively furthered in a knowledge management environment populated exclusively by learning software agents. Indeed, when simulation agents assuming central advisee and advisor knowledge management functions are placed into the same task environment and can access exposed services, a short-term knowledge network is created dynamically for the scope of a single adaptation episode.

The macro evaluation for both domains thereby confirmed a reasonably high chance for a positive real-world effect when simulating the operationalisation of an individual advisee classification model that has been modified in an adap-

tation episode. While the evaluation cautioned that an improvement cannot be guaranteed and in case of success, improvements brought about by a single adaptation episode tend to be moderate, it has been confirmed quantitatively that meaningful knowledge creation is brought about, with potential enhancements on the horizon in future work. To conclude, multiagent interactive adaptation of individual classification models has been confirmed to afford knowledge creation among learning agents in dynamic knowledge networks.

**Assessment from the Machine Learning Perspective** From the machine learning perspective the research question to be answered by the conducted empirical evaluation was whether the technique of argument based machine learning could effectively transferred from an agent-human setting into an all-agent setup. The data presented throughout the evaluation of the experiment series for both task domains demands a differentiated answer to this research question. In the first instance, the transition to the all-agent setting has been effective. Quantifiable improvements to the trained classifiers of novice agents with very restricted exposure to their task domain in terms of the absolute number of instances to learn from have been confirmed in the macro-evaluation for both investigated domains, albeit with typically moderate gains in simulated operationalisation, that is based on the performance measurements with a global benchmark dataset as frame of reference.

Also, it has been shown in Section 7.4.2 on page 166 ff. that the experimental setup in the empirical study induces a considerable fluctuation of adaptation results across similar consecutive experiment runs. However, the empirical study itself located the issue, namely the scope advisee test set used for internal advisee measurements to control its adaptation process, and identified one possible improvement to the design of the dynamic knowledge network.

As part of the empirical evaluation, it has been shown paradigmatically in Section 7.5.3 that in knowledge network configurations with heterogeneous advisor panels, interactive multiagent adaptation in the investigated implementation could not reap the full potential advisor expertise during the observed adaptation episodes. Specifically, the level of high-performing advisors could not be approximated. Instead, improvements were constrained to those of lower-performing advisors. The assessment of the experiment series suggests a correlation with advisor inference techniques. So, while the experiment series discussed in this chapter have treated the induction algorithm of the advisor panel as a variable of the experiments that constitute a complete experiment series, thus addressing the topic of heterogeneity as potential driver/inhibitor of the the desired adaptation directly, the results still demand further investigation in the future.

The experiment series for both task domains purveyed quantitative evidence to the fact that a mixture of learning algorithms would improve on comparable homogeneous setups (cf. Section 7.4.2). However, the data also showed that the performance of advisors using a specific learning algorithm in experiments with homogeneous advisors panels did not necessarily correlate with the predictive accuracy of the advisor classifiers themselves. The empirical study thus raised awareness that supplemental experiments will be necessary to better characterise key characteristics of an advisor model that promote or demote contributions to the adaptation methodology at hand. One further question that is equally of

interest for future investigation is whether the ABCN2 approach employed by the advisee to integrate advice for specific learning problem poses a limiting factor.

Concluding, from the machine learning perspective, the empirical study presented in this chapter has confirmed that the proposed adaptation methodology is effective. It has further led to insights pointing at future optimisation potentials.

**Assessment from the Technology Perspective** From the technology perspective, the empirical evaluation has confirmed that the modular architectural approach for a flexible integration of heterogeneous induction techniques implemented by means of different machine learning frameworks and programming languages held up well throughout the conducted study. What is more, it can be projected that the framework architecture that has been implemented in terms of a PlaSMA scenario provides sufficient extension points to support future extensions that will be discussed as part of the outlook of this thesis.

Overall, it can be concluded that the empirical study has confirmed the general validity of interactive multiagent adaptation of individual classification models. Beyond serving this primary objective, the empirical study has also been instrumental in the identification of optimisation potentials that point to extensions and modifications of the investigated adaptation approach while staying within the architectural and methodical framework introduced in this thesis.

Chapter 8

# Conclusions and Future Work

Autonomous process control that is implemented by means of intelligent agents in open multiagent systems presupposes access to adequate knowledge as foundation for informed situation assessment and decision making. Agent-oriented knowledge management envisions a comprehensive set of knowledge management functions for individual information retrieval and knowledge creation. Focussing on knowledge creation in the induction of classification models, flexibility with regard to applicable creation and advancement of induced models constitutes a competitive advantage for the learning agent. Such flexibility is enabled in particular by the ability to incorporate relevant knowledge distributed among the agent community of practice to improve and shape individual induction.

To that end, this thesis has developed a comprehensive methodology and an implementation for interactive knowledge creation for intelligent agents within the framework of agent-oriented knowledge management. The empirical study in the preceding chapter that was based on two independent task domains provided the basis for an extensive critical assessment of the proposed methodology. It validated the initial research hypothesis that was expressed in the introduction. In addition, the study allowed for a differentiated view on the characteristics of the methodology and viable future optimisation potentials.

Section 8.1 summarises the contributions that have been compiled throughout this project. Subsequently, Section 8.2 addresses possible directions for future work.

## 8.1 Critical Assessment of Achievements and Results

The incipient motivation of this thesis employed the scenario of intelligent, learning agents whose self-sufficient means to modify and thus improve essential dynamic knowledge such as classification models for their respective decision-making processes as their timely operationalisation had proven ineffective in their momentary situation, for instance due to an inability to acquire additional training data in time.

It has been argued, that for an agent equipped with the meta-reasoning capability to recognise said circumstance, its deployment in a multiagent system as part of a community of practice - agents with similar domain tasks building on similar knowledge - could lead to flexibility in terms of novel, inherently interactive forms of knowledge creation.

It became evident that the contributions of this thesis emerges from an integration of ideas from three essential fields of research, namely multiagent systems, knowledge management, and machine learning.

Chapters 2 through 4 consequently explored these fields in search of theoretical and practical constituents for an inclusive framework for agent-oriented knowledge management. Chapter 2 recalled architectural fundamentals of learning agents and cooperation in multiagent systems. It also investigated multiagent based simulation as an adequate approach for an empirical evaluation of highly dynamic scenarios with many independent actors. Chapter 3 has been instrumental to understand knowledge management and specifically its significance for the creation of dynamic knowledge networks made up entirely from intelligent agents. Chapter 4 on machine learning identified active learning as a promising technological starting point for discourse-based learning from peers.

Integrating these constituents, Chapter 5 consequently developed the concept for a discourse-based adaptation of individual dynamic knowledge, specifically classification models. It builds up on extensive interaction with knowledgeable peers within a multiagent system. The adaption problem has been modelled as local search problem in the space of learning hypotheses reachable by iterative integration of learning advice from interaction partners. Previous work on interacting learning has been incorporated and refined for use in multiagent environments.

The concept also contributed integral parts to enable the goal-oriented formation of dynamic knowledge networks. These comprise mechanisms for discoverability and enlisting of agents as knowledgeable interaction partners, i.e., experts for a specific learning task.

Chapter 6 fleshed out the conceptual approach for the proposed adaptation methodology with a reference implementation in the PlaSMA multiagent based simulation environment. This entailed the development of a flexible role-based architecture for JADE agents and a comprehensive integrated learning subsystem and knowledge model maintenance akin to version control systems.

Chapter 7 introduced a comprehensive test environment for the empirical assessment of this integrated adaptation approach with different compositions of agents and learning input. This test environment has been implemented on top of the general-purpose multiagent-based simulation environment PlaSMA. This contribution enabled conducting of an extensive empirical study of the reference implementation of interactive multiagent adaptation of classification models (cf. Chapter 6). The study in two application domains informed a detailed evaluation that validated the proposed adaptation concept and thus confirmed the initial research hypothesis that posited that *under qualified conditions, both the predictive accuracy and appropriation time including data acquisition of individual dynamic knowledge used in decision making can be improved when a learning agent is equipped with effective means to leverage empirical dynamic*

*knowledge of suitable non-adversarial interaction partners in a multiagent system.* It has been substantiated that *this goal can be accomplished in a distributed approach to knowledge transfer which retains the agents' responsibility for their own learning tasks and does not require the centralisation of empirical data.* Finally, the experimental design has highlighted promising directions for future research.

## 8.2 Directions for Future Research

The presented research is fundamental to a comprehensive methodical and application-oriented approach to exploit cooperation in the pursuit of individual knowledge creation within dynamic knowledge networks formed by learning, intelligent software agents.

Based on the existing body of work, several directions for complementary future research can be identified. These directions are highlighted below. They are derived from findings acquired throughout this thesis and in particular the empirical study presented in Chapter 7.

Section 8.2.1 begins with a discussion of open research questions that can be answered by means of modifications of and extensions to the experiment design in Chapter 7. Section 8.2.2 then broaches the establishment of a broader, community-sourced basis for model valuation in adaptation episodes.

The next two research directions seek to complement and enhance the proposed methodology for the adaptation of individual classification models. Section 8.2.4 argues for the investigation of benefits imposed by different forms of heterogeneity including knowledge representation and domain representation across knowledge network stakeholders and potential challenges with regard to their interoperability. Section 8.2.3 proposes an evolution of the applied adaptation control schemes. This evolution comprises for instance the application of heuristic search and enhanced advice processing to determine potential efficiency gains and a more comprehensive search space exploration.

Both the conducted experimental study in Chapter 7 and the continuations mentioned thus far focused on the conduct of single adaptation episodes corresponding to isolated situational settings to be found in a comprehensive multiagent application. Once the immediate benefits of interactive multiagent adaptation of classification models for single adaptation episodes are understood, the logical next step discussed in Section 8.2.5 is to have the presented knowledge management system operationalised continuously in a complete multiagent application. This enables empirical studies focussing on the sustained, on-demand formation of dynamic knowledge networks and their strategic impact for the individual agent and the multiagent system as a whole.

### 8.2.1 Supplements to the Empirical Study

While the experimental setup for the empirical study in Section 7.2.2 has been appropriate to answer the research questions posed in Section 7.1, supplemental aspects relevant to a complete empirical treatment of interactive multiagent adaptation of individual classification models should be investigated in concerted variations of the presented experimental setup.

**Adaptation Contribution Tracing with Performance Indicators** The assessment of results of the conducted experiments have revealed limitations in the ability to trace respective *contributions* of members of an advisor panel throughout an adaptation episode. Sections 7.4.2 and 7.5.2 have thus been inhibited in the conduct of detailed contribution analyses that would have been instrumental to better elaborate the positive effect of algorithmic heterogeneity in advisor panels that has been confirmed quantitatively for both investigated domains.

**Additional Task Domains** Section 7.3 highlighted practical contributions of the presented research towards a streamlined appropriation of task domains and matching data repositories for learning experiments. These have been essential to conduct the empirical study in Chapter 7 on two real-world domains, both validating the proposed knowledge adaptation methodology on a broader basis and finding characteristic traits of the methodology.

This work should be continued for further task domains characterised by rich categorical and numerical feature vectors. Given that the studies in Section 7.4 and Section 7.5 both found the investigated adaptation methodology to yield moderate improvements in predictive accuracy, a broadening of investigated application scenarios is likely to allow for conclusions on the influence of domain idiosyncrasies on adaptation performance. In case variations can be demonstrated, follow-up research questions emerge, specifically, which characteristics influence the effectiveness of the presented adaptation methodology.

**The Role of Advisor Multiplicity** The empirical studies on the Fars 2011 Motorist dataset in Section 7.4 and the Neota land covertype dataset in Section 7.5 treated the size of the advisor panel interacting with the respective advisee as an invariant parameter. Six advisors have been active throughout the experiments (cf. Section 7.2.2). A systematic variation of advisor multiplicity can address whether a larger body of experience codified in contributing advisor classifiers translates to a discernible positive effect on the effectiveness of single adaptation episodes. Such an investigation would yield valuable insight towards the development of a heuristic for advisor panel sizing within a given task domain (cf. Section 6.3.1).

The conduct of such experiments across an assortment of domains would further contribute to an understanding regarding the interrelation between task domain and the effective scope of the investigated dynamic knowledge networks.

**The Potential of Specialist Advisors** The empirical study on the FARS 2011 Motorist dataset in Section 7.4 showed that even the members of the advisor panel in adaptation episodes had issues in discriminating well enough among the four possible target classes, i.e., injury severities. A related research question not addressed in Chapter 7 concerns potential benefits of specialist advisors whose classifiers are tuned to an improved discrimination of specific, typically non-majority classes.

Specialist advisors can be modelled as working on a minority class and its dichotomous class. Illustrating the strengths of the experiment framework presented in Section 7.3, it is conceivable to introduce the desired advisor diversity with the existing technological basis as follows. Global dichotomy datasets can be derived from the original dataset by means of data preprocessing or the creation of dedicated database views. Experiment configurations with specialist advisors could then use multiple dedicated warehouse agents each handling one data repository, be it all-classes or dichotomous. Learners identify their matching data warehouse to acquire learning data. In addition, the distribution of real and dichotomous classes can be controlled in detail through the experiment configuration. Thus, without the need to modify agent code, the experimental framework affords a supplementary investigation on the effects of mixing generalists and specialists in the advisor panel created for adaptation episodes.

### 8.2.2 Community-driven Basis of Valuation for Adaptation

Section 5.2.4 proposed the concept of a dedicated knowledge management role to provide an independent and trustworthy model benchmarking service.

This role has not been implemented as part of the reference implementation of the adaptation framework described in Chapter 6. The rationale was that the performance assessment capability procured by this secondary knowledge management role was to facilitate primarily the opening phase of a model adaptation episode providing insight when and with whom to engage as advisee. However, as described in Section 7.4.2, the empirical study has revealed that the advisee valuation of intermediate classifiers *throughout* an adaptation episode was indeed often inaccurate due to a too small test set as the frame of reference. The use of the benchmark service role in these situations promises to mitigate such inaccuracies using a representative frame of reference in the guise of a community-sourced dataset. Hence, further research should flesh out the details of the benchmark service and provide a reference implementation.

The research challenge is the establishment and maintenance of a trustworthy classifier benchmark service with access to representative community-sourced data while at the same time guaranteeing the necessary degree of data privacy. Specifically, participants in dynamic knowledge networks need a credible incentive to procure reference data. Such an incentive could be the right to utilise benchmark services, be it benchmarking of a classifier, or access to knowledge brokering functions.

An extended version of the dynamic knowledge networks investigated in Chapter 7 could be evaluated with the same experimental setup, allowing for a direct

differential assessment.

### 8.2.3 Evolution of Adaptation Control and Advice Processing

As part of the presentation of the concept of interactive multiagent adaptation of individual classification models in Chapter 5, Section 5.3 modelled adaptation as an online search problem. Consequently, while discussing the practical implementation of the proposed approach in Chapter 6, Section 6.3.1 underscored that said online search has been realised in the reference implementation in terms of a hill climbing search.

**Online Search Algorithm** As the empirical study in Chapter 7 has proven for two independent task domains, this algorithmic baseline version for the exploration of the search space of modified classification models that are reachable from an initial model using the available operator for state transitions already sufficed to prove the fundamental validity of the presented approach to knowledge creation for intelligent agents (cf. Section 7.6). Given this initial success, it has thus become worthwhile to extend the focus of future research to the adoption of suitable, more sophisticated online search algorithms and expansion heuristics.

Such research would seek to mitigate the shallow search traces observed in the empirical study (cf. Figure 7.15 on page 177 for the FARS dataset and Figure 7.36 on page 208 for the land covertype dataset) and the total cost of successor state expansions (cf. Section 5.3.1), specifically for those with a considerable branching factor (cf. Section 7.4.2).

#### Richer Argument Advice Compilation

As for the argument advice that constitutes the acquired additional background knowledge driving search state expansion via model re-learning, only positive arguments have been supported thus far. As a consequence, the full expressiveness of the advice language as introduced in Section 5.4.3 is yet to be utilised. To that end, the prior work such as (Ontañón & Plaza 2010*c*) can be consulted.

It is also worthwhile to consider related approaches to knowledge integration such as argumentation based multi-agent joint learning (Xu et al. 2015).

### 8.2.4 Interoperability among Adaptation Stakeholders

**Heterogeneity of knowledge representations** Chapter 1.2 called for a focus on algorithmic heterogeneity to be exploited in the proposed adaptation methodology. To that end, Section 4.1 sketched the landscape of induction techniques. Although a degree of algorithmic heterogeneity has been addressed in Section 7.2.2 and implemented as shown in Section 6.2, only the family of symbolic classification techniques has been explored.

Consequently, future work ought to pursue the support for sub-symbolic classification techniques whose performance has undisputedly surpassed symbolic ancestors. From the scientific perspective, it is interesting for several reasons to investigate the adoption of, for instance, support vector machines, deep learning models or Bayesian classifiers in concert with the existing options.

To begin with, Section 5.4 introduced an advice language for learning problems whose structure is modelled after propositional rules as used in rule induction algorithms. That language was chosen due to its white box character allowing easy inspections and due to advantages in the advice integration.

Probabilistic representations, such as Bayesian networks, and in particular sub-symbolic representations, as exemplified by the artificial neural networks employed in deep learning techniques, or support vector machines (cf. Section 4.1), posses an internal structure which requires a higher investment to transform such models into symbolic representations. Nevertheless, Section 4.1.4 substantiated that respective research has been and is still actively pursued as opaque Deep Learning approaches have become hugely popular in machine learning research.

It is interesting to investigate whether the competitive edge of sub-symbolic techniques can be effectively retained for advisory duties in interactive multiagent adaptation. Preliminary evidence in support of a positive answer to this research question has been presented in by de Fortuny & Martens (2015). A thorough empirical assessment thereby needs to account for several potential factors inhibiting the advisory performance. First, the language-level translation could induce problems. Second, the advice language may be too restrictive. Finally, the techniques employed by the advisee to process acquired learning advice may turn out to introduce a limiting factor to the effectiveness of the proposed adaptation framework.

**Heterogeneity of Domain Representations** A critical prerequisite for the establishment of dynamic knowledge networks for interactive multiagent adaptation of individual classification models is a mutual understanding with regard to the task domain and its formal specification in terms of features and target classes. In the presented work, such mutual understanding has been tacitly assumed since all adaptation stakeholders shared the same data representation. While this decision was necessary to focus on main contributions, it presents an opportunity to explore another aspect of agent-oriented knowledge management marked out in Section 3.4, namely the field of semantic mediation (Hribernik, Kramer, Hans & Thoben 2010). Contributions in that area could complement the dedicated system of essential roles for agent-oriented knowledge management that has been proposed in Section 5.3.

### 8.2.5 Ongoing Integrated Operation in an Application Domain

A common denominator of both the presented experimental study on interactive multiagent adaptation of classification models in Chapter 7 and the directions for future work discussed thus far is the research focus on the initiation, conduct, and assessment of *single* adaptation episodes. Section 7.2.2 specifically highlights

this circumstance.

In particular, the extensions to the PlaSMA multiagent simulation environment presented in Section 7.3 specifically address the tailored setup of initial situations for adaptation such that the experiments then correspond to temporal excerpts from the operation of a multiagent application in which the stakeholders beside primary domain roles also pursue the necessary knowledge management activities.

The integration of interactive multiagent adaptation into an actual multiagent application, for instance in the framework of autonomous cooperating logistic processes, creates interesting additional research questions.

First, as the presented knowledge adaptation framework constitutes a social knowledge creation modality that complements traditional self-sufficient learning activities, it is interesting to study the formation, reconfiguration (in terms of role adoption), and frequency of application of dynamic knowledge networks by groups of agents in an open multiagent system with a significant agent turnover. That is to say, an environment where agents enter the system successively as time passes, thus causing inequalities in the fund of experience and consequently the quality of knowledge compiled in individual learning activities that is available to agents at certain points in time. Agents would also once in a while leave the multiagent system, thus disturbing the established distribution of knowledge in the agent community. It is conceivable that further variances would be caused by internal knowledge management functions such as purposeful oblivion of obsolescent information and thereof derived knowledge.

It should be investigated whether the presented adaptation approach through continuous application acts as a regulating factor for the empirical knowledge distributed in a multiagent system. Ultimately, the approach may constitute a means to permeate empirical knowledge in a multiagent system beyond the life cycle of the agent whose knowledge creation activities originally induced that knowledge.

From an experimental point of view, a challenge will be to devise an adequate support for the ascertainment and assessment of key performance indicators in the face of the significant dynamics of the investigated scenario. However, with an adequate experimental setup in place, the investigation of the complete integration of interactive multiagent adaptation of individual classification models as originally envisioned in chapter 1 would be possible at last.

# Bibliography


Abar, S., Theodoropoulos, G. K., Lemarinier, P. & O'Hare, G. M. P. (2017), 'Agent Based Modelling and Simulation tools: A review of the state-of-art software', *Computer Science Review* **24**(Supplement C), 13–33.

Abecker, A., Bernardi, A. & Van Elst, L. (2003), Agent Technology for Distributed Organizational Memories, *in* 'Proceedings of the 5th International Conference on Enterprise Information Systems', Vol. 2, Angers, France, pp. 3–10.

Abecker, A., Hinkelmann, K., Maus, H. & Müller, H. J. (2002), *Geschäftsprozessorientiertes Wissensmanagement: Effektive Wissensnutzung bei der Planung und Umsetzung von Geschäftsprozessen*, Springer Verlag Berlin / Heidelberg.

Ackoff, R. L. (1989), 'From Data to Wisdom', *Journal of Applied Systems Analysis* **16**, 3–9.

Alavi, M. & Leidner, D. E. (2001), 'Review: Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues', *MIS quarterly* pp. 107–136.

Augasta, M. G. & Kathirvalavakumar, T. (2012), Rule extraction from neural networks — A comparative study, *in* 'International Conference on Pattern Recognition, Informatics and Medical Engineering (PRIME-2012)', pp. 404–408.

Barakat, N. H. & Bradley, A. P. (2007), 'Rule Extraction From Support Vector Machines: A Sequential Covering Approach', *IEEE Transactions on Knowledge and Data Engineering* **19**(6), 729–741.

Bechhofer, S., van Hamelen, F., Hendler, J. & Others (2004), OWL Web Ontology Language Reference, W3C Recommendation, Technical report, W3C.

Bellifemine, F., Bergenti, F., Caire, G. & Poggi, A. (2005), Jade — A Java Agent Development Framework, *in* R. H. Bordini, M. Dastani, J. Dix & A. El Fallah Seghrouchni, eds, 'Multi-Agent Programming: Languages, Platforms and Applications', Springer US, Boston, MA, pp. 125–147.

Bellifemine, F. L., Caire, G. & Greenwood, D. (2007), *Developing Multi-Agent Systems with JADE*, Vol. 5 of *Wiley Series in Agent Technology*, Wiley.

Bench-Capon, T. & Dunne, P. (2007), 'Argumentation in Artificial Intelligence', *Artificial Intelligence* **171**(10-15), 619–641.

Bergenti, F., Caire, G. & Gotta, D. (2014), Agents on the Move: Jade on Android Devices, *in* C. Santorro, ed., 'Proceedings of the XV Workshop "Dagli Oggetti agli Agenti" (WOA 2014)', CEUR-WS, volume 1260, Catalina, Italy, pp. 25–26.

Bergenti, F. & Poggi, A. (2002), LEAP: A FIPA Platform for Handheld and Mobile Devices, *in* J.-J. C. Meyer, & M. Tambe, eds, 'Intelligent Agents VIII: Agent Theories, Architectures, and Languages 8th International Workshop, ATAL 2001 Seattle,WA, USA, August 1–3, 2001 Revised Papers', Springer Berlin Heidelberg, pp. 436–446.

Bergenti, F., Poggi, A. & Rimassa, G. (2000), Agent Architecture and Interaction Protocols for Corporate Memory Management Systems, *in* 'Proceedings of the ECAI 2000 Workshop on Knowledge Management and Organizational Memories', Berlin, Germany, pp. 39–47.

Berndt, J.-O. (2016), Self-organizing Multiagent Negotiations, Doctoral dissertation, Universität Bremen.

Blackard, J. A., Dean, D. J. & Others (1999), 'Comparative Accuracies of Artificial Neural Networks and Discriminant Analysis in Predicting Forest Cover Types from Cartographic Variables', *Computers and Electronics in Agriculture* **24**(3), 131–152.

Bloos, M., Warden, T., Kopfer, H. & Herzog, O. (2011), Modeling and Simulation of Operational Transport Collaboration: A Multiagent-based Approach, *in* 'Tagungsband des 6. Deutsch-Russischen Logistik Workshops', Cuvillier Verlag, Göttingen, pp. 167–179.

Blum, A. & Mitchell, T. (1998), Combining Labeled and Unlabeled Data with Co-Training, *in* 'Proceedings of the 11th Annual Conference on Computational Learning Theory - COLT' 98', ACM Press, New York, New York, USA, pp. 92–100.

Bonifacio, M., Bouquet, P., Mameli, G. & Nori, M. (2002), Kex: A Peer-to-peer Solution for Distributed Knowledge Management, *in* D. Karagiannis & U. Reimer, eds, 'Practical Aspects of Knowledge Management: 4th International Conference, PAKM 2002 Vienna, Austria, December 2-3, 2002 Proceedings', Springer Berlin / Heidelberg, pp. 490–500.

Bonifacio, M., Bouquet, P., Mameli, G. & Nori, M. (2004), Peer-mediated Distributed Knowledge Management, *in* L. van Elst, V. Dignum & A. Abecker, eds, 'Agent-Mediated Knowledge Management: International Symposium AMKM 2003, Stanford, CA, USA, March 24-26, Revised and Invited Papers', Springer Berlin / Heidelberg, pp. 31–47.

Bonifacio, M., Bouquet, P. & Traverso, P. (2002), 'Enabling Distributed Knowledge Management: Managerial and Technological Implications', *Novatica and Informatik/Informatique* **3**(1), 23–29.

Booch, G., Rumbaugh, J. & Jacobson, I. (2005), *The Unified Modeling Language User Guide*, 2 edn, Addison Wesley Longman, Amsterdam, The Netherlands.

Bordini, R. H., Braubach, L., Dastani, M., Seghrouchni, A. E. F., Gomez-Sanz, J. J., Leite, J., O'Hare, G., Pokahr, A. & Ricci, A. (2006), 'A Survey of Programming Languages and Platforms for Multi-Agent Systems', *Informatica* **30**, 33–44.

Borshchev, A. & Filippov, A. (2004), From System Dynamics and Discrete Event to Practical Agent based Modeling: Reasons, Techniques, Tools, *in* M. Kennedy, G. W. Winch, R. S. Langer, J. I. Rowe & J. M. Yanni, eds, 'Proceedings of the 22nd International Conference of the System Dynamics Society', Oxford, UK.

Böse, F. & Windt, K. (2007), Catalogue of Criteria for Autonomous Control, *in* M. Hülsmann & K. Windt, eds, 'Understanding Autonomous Cooperation and Control in Logistics – The Impact on Management, Information, Communication and Material Flow', Springer, Berlin, pp. 57–72.

Bratman, M. (1987), *Intention, Plans, and Practical Reason*, Havard University Press, Cambidge, MA, USA.

Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F. & Mylopoulos, J. (2004), 'Tropos: An Agent-oriented Software Development Methodology', *Autonomous Agents and Multi-Agent Systems* **8**(3), 203–236.

Buchanan, B. G., Mitchell, T. M., Smith, R. G. & Johnson, C. R. (1978), 'Models of Learning Systems', *Encyclopedia of Computer Science and Technology* **11**, 24–51.

Carlsson, S. A., El Sawy, O. A., Eriksson, I. & Raven, A. (1996), Gaining Competitive Advantage Through Shared Knowledge Creation: In Search of a New Design Theory for Strategic Information Systems, *in* J. Dias Coelho, T. Jelassi, W. König, H. Krcmar, O. R. & M. Sääksjarvi, eds, 'Proceedings of the 4th European Conference on Information Systems', Lisbon, Portugal, pp. 1067–1075.

Castle, C. J. E. & Crooks, A. T. (2006), Principles and Concepts of Agent-based Modelling for Developing Geospatial Simulations, Technical report, Centre for Advanced Spatial Analysis (UCL), UCL (University College London), Centre for Advanced Spatial Analysis (UCL), London, GB.

Cendrowska, J. (1987), 'PRISM: An Algorithm for Inducing Modular Rules', *International Journal of Man-Machine Studies* **27**(4), 349–370.

Chaib-draa, B. & Müller, J., eds (2006), *Multiagent-based Supply Chain Management*, number 28 *in* 'Studies in Computational Intelligence', Springer.

Chakraborty, D. & Sen, S. (2006), Teaching New Teammates, *in* P. Stone & G. Weiß, eds, 'Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems', AAMAS '06, ACM, New York, NY, USA, pp. 691–693.

Chakraborty, M., Biswas, S. K. & Purkayastha, B. (2018), 'Recursive Rule Extraction from NN using Reverse Engineering Technique', *New Generation Computing* **36**(2), 119–142.

Chawla, N. V. & Karakoulas, G. (2005), 'Learning from Labeled and Unlabeled Data: An Empirical Study Across Techniques and Domains', *Journal of Artificial Intelligence Research* **23**(1), 331–366.

Chen, J. R., Wolfe, S. R. & Wragg, S. D. (2000), A Distributed Multi-agent System for Collaborative Information Management and Sharing, *in* 'Proceedings of the 9th International Conference on Information and Knowledge Management', ACM, McLean, Virginia, USA, pp. 382–388.

Chorowski, J. & Zurada, J. M. (2011), 'Extracting Rules From Neural Networks as Decision Diagrams', *IEEE Transactions on Neural Networks* **22**(12), 2435–2446.

Clark, P. & Boswell, R. (1991), Rule Induction with CN2: Some Recent Improvements, *in* Y. Kodratoff, ed., 'Machine Learning — EWSL-91', Vol. 482 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 151–163.

Clark, P. & Niblett, T. (1989), 'The CN2 Induction Algorithm', *Machine Learning* **3**(4), 261–283.

Cohen, W. W. (1995), Fast and Effective Rule Induction, *in* 'Machine Learning: Proceedings of the 12th International Conference', Morgan Kaufmann, Lake Tahoe, California, USA, pp. 115–123.

Cohn, D., Atlas, L. & Ladner, R. (1994), 'Improving Generalization with Active Learning', *Machine Learning* **15**(2), 201–221.

Costantini, S. & Tocchio, A. (2005), Learning by Knowledge Exchange in Logical Agents, *in* F. Corradini, F. De Paoli, E. Merelli & A. Omicini, eds, 'Proceedings of the 6th AI*IA/TABOO Joint Workshop "From Objects to Agents", WOA', Pitagagora Editrice, Bologna, Italy, pp. 1–8.

Craig, I. (2000), *The Interpretation of Object-Oriented Programming Languages*, Springer, London.

Craven, M. W. & Shavlik, J. W. (1996), 'Extracting Tree-structured Representations of Trained Networks', *Advances in Neural Information Processing Systems* pp. 24–30.

Cristianini, N., Shawe-Taylor, J. & Others (2000), *An introduction to support vector machines and other kernel-based learning methods*, Cambridge university press.

Curk, T., Demšar, J., Xu, Q., Leban, G., Petrovič, U., Bratko, I., Shaulsky, G. & Zupan, B. (2005), 'Microarray Data Mining with Visual Programming', *Bioinformatics* **21**(3), 396–398.

Dale, J. & Mamdani, E. (2001), 'Open Standards for Interoperating Agent-based Systems', *Software Focus* **2**(1), 1–8.

Davenport, T. H. & Prusak, L. (2000), *Working Knowledge: How Organizations Manage what They Know*, Harvard Business Press, Boston, MA, USA.

Davidsson, P. (2001), Multi Agent Based Simulation: Beyond Social Simulation, *in* S. Moss & P. Davidsson, eds, 'Multi-Agent-Based Simulation: Second International Workshop, MABS 2000 Boston, MA, USA, July Revised and Additional Papers', Springer Berlin / Heidelberg, pp. 97–107.

Davidsson, P., Holmgren, J., Kyhlbäck, H., Mengistu, D. & Persson, M. (2007), Applications of Agent Based Simulation, *in* L. Antunes & K. Takadama, eds, 'Multi-Agent-Based Simulation VII', Vol. 4442 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 15–27.

Davidsson, P., Holmgren, J., Persson, J. A. & Ramstedt, L. (2008), Multi Agent Based Simulation of Transport Chains, *in* 'Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2', AAMAS '08, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp. 1153–1160.

de Fortuny, E. J. & Martens, D. (2015), 'Active Learning-Based Pedagogical Rule Extraction', *IEEE Transactions on Neural Networks and Learning Systems* **26**(11), 2664–2677.

Dignum, V. (2004), A Model for Organizational Interaction: Based on Agents, Founded in Logic., Doctoral dissertation, Utrecht University, Utrecht, The Netherlands.

Dignum, V. (2006), 'An Overview of Agents in Knowledge Management', *Declarative Programming for Knowledge Management* pp. 175–189.

Druck, G., Mann, G. & McCallum, A. (2008), Learning from Labeled Features Using Generalized Expectation Criteria, *in* 'Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '08', ACM, Singapore, Singapore, pp. 595–602.

Durfee, E. H. (2001), Distributed Problem Solving and Planning, *in* M. Luck, ed., 'Multi-Agent Systems and Applications: 9th ECCAI Advanced Course, ACAI 2001 and Agent Link's 3rd European Agent Systems Summer School, EASSS 2001 Prague, Czech Republic, July 2–13, 2001 Selected Tutorial Papers', Springer Berlin / Heidelberg, pp. 118–149.

Durstenfeld, R. (1964), 'Algorithm 235: Random Permutation', *Communications of the ACM* **7**(7), 420.

Edelkamp, S., Greulich, C., Gath, M., Warden, T., Human, M., Herzog, O. & Sitharam, T. G. (2013), Enhanced Shortest Path Computation for Multiagent-based Intermodal Transport Planning in Dynamic Environments, *in* J. Filipe & A. Fred, eds, '5th International Conference on Agents and Artificial Intelligence', SciTePress, Barcelona, Spain, pp. 324–329.

Ester, M., Kriegel, H. P., Sander, J. & Xu, X. (1996), A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, *in* E. Simoudis, J. Han & U. Fayyad, eds, 'Proceedings of the 2nd International Conference on Knowledge Discovery and Data mining', Vol. 1996, AAAI Press, Menlo Park, CA, USA, pp. 226–231.

Ester, M. & Sander, J. (2000), *Knowledge Discovery in Databases: Techniken und Anwendungen*, Springer Berlin / Heidelberg.

Fayyad, U., Piatetsky-Shapiro, G. & Smyth, P. (1996), 'From Data Mining to Knowledge Discovery in Databases', *AI magazine* **17**(3), 37.

Ferber, J. (2001), *Multiagentensysteme: Eine Einführung in die verteilte künstliche Intelligenz*, Addison-Wesley.

Fischer, G. & Otswald, J. (2001), 'Knowledge Management: Problems, Promises, Realities, and Challenges', *Intelligent Systems, IEEE* **16**(1), 60–72.

Fischer, K., Schillo, M. & Siekmann, J. (2003), Holonic Multiagent Systems: A Foundation for the Organisation of Multiagent Systems, *in* V. Marík, D. McFarlane & P. Valckenaers, eds, 'Holonic and Multi-Agent Systems for Manufacturing', Vol. 2744 of *Lecture Notes in Computer Science*, Springer Berlin/Heidelberg, pp. 1083–1084.

Fisher, R. A. (1936), 'The use of multiple measurements in taxonomic problems', *Annals of eugenics* **7**(2), 179–188.

Fishman, G. (2013), *Discrete-event Simulation: Modeling, Programming, and Analysis*, Springer Verlag New York.

Forrester, J. W. (1961), *Industrial Dynamics*, MIT Press, Cambridge, MA, USA.

Foundation for Intelligent Physical Agents (2001*a*), 'FIPA Dutch Auction Interaction Protocol Specification', *(Document No. XC00032F)* .

Foundation for Intelligent Physical Agents (2001*b*), 'FIPA English Auction Interaction Protocol Specification', *(Document No. XC00031F)* .

Foundation for Intelligent Physical Agents (2002*a*), 'FIPA Abstract Architecture Specification', *(Standard No. SC00001L)* .

Foundation for Intelligent Physical Agents (2002*b*), 'FIPA Brokering Interaction Protocol Specification', *(Standard No. SC00033H)* .

Foundation for Intelligent Physical Agents (2002*c*), 'FIPA Contract Net Interaction Protocol Specification', *(Standard No. SC00029H)* .

Foundation for Intelligent Physical Agents (2002*d*), 'FIPA Iterated Contract Net Interaction Protocol Specification', *(Standard No. SC00030H)* .

Foundation for Intelligent Physical Agents (2002*e*), 'FIPA Propose Interaction Protocol Specification', *(Standard No. SC00036H)* .

Foundation for Intelligent Physical Agents (2002*f*), 'FIPA Query Interaction Protocol Specification', *(Standard No. SC00027H)* .

Foundation for Intelligent Physical Agents (2002*g*), 'FIPA Recruiting Interaction Protocol Specification', *(Standard No. SC00034H)* .

Foundation for Intelligent Physical Agents (2002*h*), 'FIPA Request Interaction Protocol Specification', *(Standard No. SC00026H)* .

Foundation for Intelligent Physical Agents (2002*i*), 'FIPA Request When Interaction Protocol Specification', *(Standard No. SC00028H)* .

Foundation for Intelligent Physical Agents (2002*j*), 'FIPA Subscribe Interaction Protocol Specification', *(Standard No. SC00035H)* .

Foundation for Intelligent Physical Agents (2004), 'FIPA Agent Management Specification', *(Standard No. SC00023K)* .

Frank, A. & Asuncion, A. (2010), 'UCI Machine Learning Repository'.
**URL:** *http://archive.ics.uci.edu/ml (Last visited: 2017-05-28)*

Freitag, M., Herzog, O. & Scholz-Reiter, B. (2004), 'Selbststeuerung logistischer Prozesse– ein Paradigmenwechsel und seine Grenzen', *Industrie Management* **20**(1), 23–27.

Freitas, A. A. (2014), 'Comprehensible Classification Models: A Position Paper', *ACM SIGKDD Explorations Newsletter* **15**(1), 1–10.

Fung, G., Sandilya, S. & Rao, R. B. (2005), Rule Extraction from Linear Support Vector Machines, *in* 'Proceeding of the 11th ACM SIGKDD International Conference on Knowledge discovery in data mining - KDD '05', ACM Press, New York, New York, USA, pp. 32–40.

Fürnkranz, J. (1999), 'Separate-and-Conquer Rule Learning', *Artificial Intelligence Review* **13**(1), 3–54.

Ganji, F., Morales Kluge, E. & Scholz-Reiter, B. (2010), Bringing Agents into Application: Intelligent Products in Autonomous Logistics, *in* K. Schill, B. Scholz-Reiter & L. Frommberger, eds, 'Artificial Intelligence and Logistics (AiLog) - Workshop at ECAI 2010', pp. 37–42.

Gath, M. (2016), *Optimizing Transport Logistics Processes with Multiagent Planning and Control*, Springer Vieweg.

Gehrke, J. D. (2011), Relevanzbasierte Informationsbeschaffung für die informierte Entscheidungsfindung intelligenter Agenten, Doctoral dissertation, Universität Bremen.

Gehrke, J. D., Herzog, O., Langer, H., Malaka, R., Porzel, R. & Warden, T. (2010), 'An Agent-Based Approach to Autonomous Logistic Processes', *Künstliche Intelligenz* **24**(2), 137–141.

Gehrke, J. D. & Ober-Blöbaum, C. (2007), Multiagent-based Logistics Simulation with PlaSMA, *in* R. Koschke, O. Herzog, K.-H. Rödiger & M. Ronthaler, eds, 'Informatik 2007. Informatik trifft Logistik. Beiträge der 37. Jahrestagung der Gesellschaft für Informatik', Köllen Druck+Verlag GmbH, Bremen, Germany, pp. 416–419.

Gehrke, J. D. & Schuldt, A. (2009), Incorporating Knowledge about Interaction for Uniform Agent Design for Simulation and Operation, *in* K. S. Decker, J. S. Sichman, C. Sierra & C. Castelfranchi, eds, 'Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-09)', International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), Budapest, Hungary, pp. 1175–1176.

Gehrke, J. D., Schuldt, A. & Werner, S. (2008), Quality Criteria for Multiagent-Based Simulations with Conservative Synchronisation, *in* M. Rabe, ed., '13th ASIM Dedicated Conference on Simulation in Production and Logistics', Fraunhofer IRB Verlag, Stuttgart, Germany, pp. 545–554.

Gehrke, J. D. & Wojtusiak, J. (2008*a*), A Natural Induction Approach to Traffic Prediction for Autonomous Agent-based Vehicle Route Planning, Technical report, MLI Laboratory, George Mason University, Fairfax, VA, USA.

Gehrke, J. D. & Wojtusiak, J. (2008*b*), Traffic Prediction for Agent Route Planning, *in* M. Bubak, G. D. van Albada, J. Dongarra & P. M. A. Sloot, eds, '8th International Conference on Computational Science 2008, vol. 3', Springer, Kraków, Poland, pp. 692–701.

Goldman, S. A. & Zhou, Y. (2000), Enhancing Supervised Learning with Unlabeled Data, *in* 'Proceedings of the 17th International Conference on Machine Learning (ICML 2000)', Stanford, CA, USA, pp. 327–334.

González, C., Loza Mencía, E. & Fürnkranz, J. (2017), Re-training Deep Neural Networks to Facilitate Boolean Concept Extraction, *in* A. Yamamoto, T. Kida, T. Uno & T. Kuboyama, eds, 'Discovery Science', Springer International Publishing, Cham, pp. 127–143.

González-Vélez, H., Mier, M., Julià-Sapé, M., Arvanitis, T. N., García-Gómez, J. M., Robles, M., Lewis, P. H., Dasmahapatra, S., Dupplaw, D., Peet, A. & Others (2009), 'HealthAgents: Distributed Multi-agent Brain Tumor Diagnosis and Prognosis', *Applied Intelligence* **30**(3), 191–202.

Goodfellow, I., Bengio, Y., Courville, A. & Bengio, Y. (2016), *Deep learning*, Vol. 1, MIT press Cambridge.

Gorodkin, J. (2004), 'Comparing two K-category Assignments by a K-category Correlation Coefficient', *Computational Biology and Chemistry* **28**(5-6), 367–374.

Graça, P. R. & Gaspar, G. (2004), Interactive Multi-agent Learning, *in* 'AISB 2004 Convention: Motion, Emotion and Cognition', Leeds, UK, pp. 95–100.

Graves, A. (2012), *Supervised Sequence Labelling with Recurrent Neural Networks*, 1 edn, Springer Berlin / Heidelberg.

Greenwood, D. & Calisti, M. (2004), Engineering Web service - Agent Integration, *in* '2004 IEEE International Conference on Systems, Man and Cybernetics', Vol. 2, IEEE, The Hague, Netherlands, pp. 1918–1925.

Greulich, C., Edelkamp, S. & Gath, M. (2013), *Agent-Based Multimodal Transport Planning in Dynamic Environments*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 74–85.

Gubbi, J., Buyya, R., Marusic, S. & Palaniswami, M. (2013), 'Internet of Things (IoT): A vision, Architectural Elements, and Future Directions', *Future Generation Computer Systems* **29**(7), 1645–1660.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F. & Pedreschi, D. (2018), 'A Survey of Methods for Explaining Black Box Models', *ACM Computing Surveys* **51**(5), 93:1–93:42.

Guizzardi, R., Aroyo, L. & Wagner, G. (2004), Agent-oriented Knowledge Management in Learning Environments: A Peer-to-Peer Helpdesk Case Study, *in* L. van Elst, V. Dignum & A. Abecker, eds, 'Agent-Mediated Knowledge Management: International Symposium AMKM 2003, Stanford, CA, USA, March 24-26, Revised and Invited Papers', Springer, pp. 57–72.

Guizzardi, R. S. S. (2006), Agent-oriented Constructivist Knowledge Management, Doctoral dissertation, University of Twente, Enschede.

Gupta, A. K. & Govindarajan, V. (2000), 'Knowledge Flows within Multinational Corporations', *Strategic Management Journal* **21**(4), 473–496.

Hahn, J. & Subramani, M. R. (2000), A Framework of Knowledge Management Systems: Issues and Challenges for Theory and Practice, *in* 'Proceedings of the 21st International Conference on Information Systems', Association for Information Systems, Brisbane, Queensland, Australia, pp. 302–312.

Hailesilassie, T. (2016), 'Rule Extraction Algorithm for Deep Neural Networks: A Review', *International Journal of Computer Science and Information Security* **14**(7), 371—-381.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. & Witten, I. H. (2009), 'The WEKA Data Mining Software: An Update', *ACM SIGKDD Explorations Newsletter* **11**(1), 10–18.

Haugen, Ø. & Runde, R. K. (2009), Enhancing UML to Formalize the FIPA Agent Interaction Protocol, *in* K. Fischer, J. P. Müller, J. Odell & A. J. Berre, eds, 'Agent-Based Technologies and Applications for Enterprise Interoperability: International Workshops, ATOP 2005 Utrecht, The Netherlands, July 25-26, 2005, and ATOP 2008, Estoril, Portugal, May 12-13, 2008, Revised Selected Papers', pp. 154–173.

He, M. & Leung, H.-f. (2002), 'Agents in E-Commerce: State of the Art', *Knowledge and Information Systems* **4**(3), 257–282.

Hellenschmidt, M. & Wichert, R. (2007), Selbstorganisation: Dinge in eigenverantwortlicher Kooperation—eine Systemanalyse, *in* 'Internet der Dinge', Springer Berlin / Heidelberg, pp. 91–105.

Henesey, L., Davidsson, P. & Persson, J. A. (2008), Agent based Simulation Architecture for Evaluating Operational Policies in Transshipping Containers, *in* 'Autonomous Agents and Multi-Agent Systems', Vol. 18, Springer Netherlands, pp. 220–238.

Herrler, R. & Klügel, F. (2006), Simulation, *in* S. Kirn, O. Herzog, P. Lockemann & O. Spaniol, eds, 'Multiagent Engineering: Theory and Applications in Enterprises', International Handbooks on Information Systems, Springer, Berlin, Heidelberg.

Herrmann, T., Jahnke, I. & Loser, K. U. (2004), The Role Concept as a Basis for Designing Community Systems, *in* F. Darses, R. Dieng, C. Simone & M. Zackland, eds, 'Cooperative Systems Design, Scenario-Based Design of Collaborative Systems', IOS Press, Amsterdam, The Netherlands, pp. 163–178.

Hribernik, K., Kramer, C., Hans, C. & Thoben, K.-D. (2010), A Semantic Mediator for Data Integration in Autonomous Logistics Processes, *in* K. Poppelwell, J. Harding, R. Poler & R. Chalmeta, eds, 'Enterprise Interoperability IV. Making the Internet of the Future for the Future of Enterprise', Springer, London, pp. 157–167.

Hribernik, K., Warden, T., Thoben, K.-D. & Herzog, O. (2010), An Internet of Things for Transport Logistics - An Approach to Connecting the Information and Material Flows in Autonomous Cooperating Logistics Processes, *in* 'Proceedings of the 12th International MITIP Conference on Information Technology & Innovation Processes of the Enterprises', Aalborg, Denmark, pp. 54–67.

Hruschka, E. R. J., do Carmo Nicoletti, M., de Oliveira, V. A. & Bressan, G. M. (2007), Markov-Blanket Based Strategy for Translating a Bayesian Classifier into a Reduced Set of Classification Rules, *in* '7th International Conference on Hybrid Intelligent Systems (HIS 2007)', IEEE, Kaiserslautern, Germany, pp. 192–197.

Huget, M.-P. (2004), 'Agent UML Notation for Multiagent System Design', *IEEE Internet Computing* **8**(4), 63.

Huget, M.-P. & Odell, J. (2004), Representing Agent Interaction Protocols with Agent UML, *in* J. Odell, P. Giorgini & J. P. Müller, eds, 'Agent-Oriented Software Engineering V: 5th International Workshop, AOSE 2004, New York, NY, USA, July 19, 2004. Revised Selected Papers', Springer Berlin Heidelberg, pp. 16–30.

Huysmans, J., Setiono, R., Baesens, B. & Vanthienen, J. (2008), 'Minerva: Sequential Covering for Rule Extraction.', *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics* **38**(2), 299–309.

Jakob, M., Tožička, J. & Pěchouček, M. (2008), Collaborative Learning with Logic-Based Models, *in* K. Tuyls, A. Nowe, Z. Guessoum & D. Kudenko, eds, 'Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning: 5th, 6th, and 7th European Symposium, ALAMAS 2005-2007 on Adaptive and Learning Agents and Multi-Agent Systems, Revised Selected Papers', Springer Berlin Heidelberg, pp. 102–116.

Jedermann, R., Behrens, C., Westphal, D. & Lang, W. (2006), 'Applying Autonomous Sensor Systems in Logistics; Combining Sensor Networks, RFIDs and Software Agents', *Sensors and Actuators A (Physical)* **132**(1), 370–375.

Jennex, M. E. (2009), Re-Visiting the Knowledge Pyramid, *in* R. H. Sprague, ed., 'Proceedings of the 42nd Hawaii International Conference on System Sciences, 2009.', IEEE, CPS, Waikoloa, Big Island, HI, USA, pp. 1–7.

Jennings, N. R. (2001), 'An Agent-based Approach for Building Complex Software Systems', *Communications of the ACM* **44**(4), 35–41.

Jünemann, R., Daum, M., Piepel, U. & Schwinning, S. (1989), *Materialfluss und Logistik: Systemtechnische Grundlagen mit Praxisbeispielen*, Springer Berlin Heidelberg.

Jurman, G., Riccadonna, S. & Furlanello, C. (2012), 'A Comparison of MCC and CEN Error Measures in Multi-Class Prediction', *PLoS ONE* **7**(8), e41882.

Kazakov, D. & Kudenko, D. (2001), Machine Learning and Inductive Logic Programming for Multi-agent Systems, *in* M. Luck, V. Mařík, O. Štěpánková & R. Trappl, eds, 'Multi-Agent Systems and Applications: 9th ECCAI Advanced Course, ACAI 2001 and Agent Link's 3rd European Agent Systems Summer School, EASSS 2001 Prague, Czech Republic, July 2–13, 2001 Selected Tutorial Papers', Springer Berlin Heidelberg, pp. 246–270.

Kerschberg, L. (1997), 'Knowledge Rovers: Cooperative Intelligent Agent Support for Enterprise Information Architectures', *Cooperative Information Agents: First International Workshop, CIA'97 Kiel, Germany, February 26–28, 1997 Proceedings* pp. 79–100.

King, R. D., Rowland, J., Oliver, S. G., Young, M., Aubrey, W., Byrne, E., Liakata, M., Markham, M., Pir, P., Soldatova, L. N., Sparkes, A., Whelan, K. E. & Clare, A. (2009), 'The Automation of Science', *Science* **324**(5923), 85–89.

Kirn, S., Herzog, O., Lockemann, P. & Spaniol, O., eds (2006), *Multiagent Engineering: Theory and Applications in Enterprises*, International Handbooks on Information Systems, Springer Berlin Heidelberg.

Kohavi, R. (1994), Bottom-up Induction of Oblivious Read-Once Decision Graphs, *in* F. Bergadano & L. De Raedt, eds, 'Machine Learning: ECML-94: European Conference on Machine Learning Catania, Italy, April 6–8, 1994 Proceedings', Vol. 764 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 154–169.

Kononenko, I. & Kukar, M. (2007), *Machine Learning and Data Mining: Introduction to Principles and Algorithms*, Horwood Publishing.

Kriegel, H.-P., Kröger, P., Sander, J. & Zimek, A. (2011), 'Density-based Clustering', *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **1**(3), 231–240.

Langer, H., Gehrke, J. D., Hammer, J., Lorenz, M. & Timm, I. J. (2005), Emerging Knowledge Management in Distributed Environments, *in* 'Proceedings of the Workshop on Agent-Mediated Knowledge Management (AMKM 2005). at the F4h International Joint Conference on Autonomous Agents & Multi Agent Systems (AAMAS 05)', Universiteit Utrecht, Utrecht, The Netherlands, pp. 14–26.

Langer, H., Gehrke, J. D., Hammer, J., Lorenz, M., Timm, I. J. & Herzog, O. (2006), 'A Framework for Distributed Knowledge Management in Autonomous Logistic Processes', *International Journal of Knowledge-Based & Intelligent Engineering Systems* **10**(4), 277–290.

Langer, H., Gehrke, J. D. & Herzog, O. (2007), Distributed Knowledge Management in Dynamic Environments, *in* M. Hülsmann & K. Windt, eds, 'Understanding Autonomous Cooperation & Control in Logistics - The Impact on Management, Information, Communication and Material Flow', Springer Berlin Heidelberg, chapter 3.5, pp. 215–231.

Lavrač, N., Flach, P. & Zupan, B. (1999), Rule Evaluation Measures: A Unifying View, *in* S. Džeroski & P. Flach, eds, 'Inductive Logic Programming: 9th International Workshop, ILP-99 Bled, Slovenia, June 24–27, 1999 Proceedings', Springer Berlin Heidelberg, pp. 174–185.

LeCun, Y., Kavukcuoglu, K., Farabet, C. & Others (2010), Convolutional Networks and Applications in Vision, *in* 'Proceedings of the 2010 IEEE International Symposium on Circuits and Systems', pp. 253–256.

Lee, J., Bagheri, B. & Kao, H.-A. (2015), 'A Cyber-physical Systems Architecture for Industry 4.0-based Manufacturing Systems', *Manufacturing Letters* **3**, 18–23.

Lees, M., Logan, B., Minson, R., Oguara, T. & Theodoropoulos, G. (2005), Distributed Simulation of MAS, *in* P. Davidsson, B. Logan & K. Takadama, eds, 'Multi-Agent and Multi-Agent-Based Simulation: Joint Workshop MABS 2004, New York, NY, USA, July 19, 2004, Revised Selected Papers', Springer Berlin Heidelberg, pp. 25–36.

Logan, B. & Theodoropoulos, G. (2001), 'The Distributed Simulation of Multiagent Systems', *Proceedings of the IEEE* **89**(2), 174–185.

Lorenzen, L., Woelk, P.-O., Denkena B., S. T., Timm, I. J. & Herzog, O. (2006), Integrated Process Planning and Production Control, *in* 'Multiagent Engineering: Theory and Applications in Enterprises', International Handbooks on Information Systems, Springer, pp. 91–113.

Luger, G. F. (2005), *Artificial intelligence: Structures and Strategies for Complex Problem Solving*, Addison-Wesley Longman.

Mann, G. & McCallum, A. (2008), Generalized Expectation Criteria for Semi-supervised Learning of Conditional Random Fields, *in* 'Proceedings of the Association for Computational Linguistics (ACL)', ACL Press.

Martens, D., Baesens, B. & Van Gestel, T. (2009), 'Decompositional Rule Extraction from Support Vector Machines by Active Learning', *IEEE Transactions on Knowledge and Data Engineering* **21**(2), 178–191.
**URL:** *http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4564457*

Massive Online Analysis (2016), 'MOA Massive Online Analysis - Datasets'.
**URL:** *http://moa.cms.waikato.ac.nz/datasets/ (Last visited: 2016-09-26)*

Maurer, H. (2003), Important Aspect of Knowledge Management, *in* R. Klein, H.-W. Six & L. Wegner, eds, 'Computer Science in Perspective: Essays Dedicated to Thomas Ottmann', Springer Berlin Heidelberg, pp. 245–254.

McQueen, R. (1998), Four Views of Knowledge and Knowledge Management, *in* 'Proceedings of the 4th Americas Conference on Information Systems', Association for Information Systems, Baltimore, MD, USA, pp. 609–611.

Melville, P. & Sindhwani, V. (2009), Active Dual Supervision: Reducing the Cost of Annotating Examples and Features, *in* E. Ringger, R. Haertel & K. Tomanek, eds, 'Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing', Omnipress Inc., Boulder, Colorado, USA, pp. 49–57.

Meyer, G. G., Främling, K. & Holmström, J. (2009), 'Intelligent Products: A Survey', *Computers in industry* **60**(3), 137–148.

Michalski, R. S. (1993), Inferential Theory of Learning as a Conceptual Basis for Multistrategy Learning, *in* R. S. Michalski, ed., 'Multistrategy Learning', Vol. 240 of *The Kluwer International Series in Engineering and Computer Science*, Springer US, pp. 3–43.

Možina, M., Guid, M., Krivec, J., Sadikov, A. & Bratko, I. (2008), Fighting Knowledge Acquisition Bottleneck with Argument Based Machine Learning, *in* M. Ghallab, C. D. Spyropoulos & N. Fakotakis, eds, 'Proceeding of the 18th European Conference on Artificial Intelligence (ECAI 2008)', IOS Press, Patras, Greece, pp. 234–238.

Možina, M., Guid, M., Krivec, J., Sadikov, A. & Bratko, I. (2010), Learning to Explain with ABML, *in* 'Proceedings of the 2010 ExaCt Workshop on Explanation-aware Computing', Lisbon, Portugal, pp. 37–48.

Možina, M., Zabkar, J. & Bratko, I. (2006), Implementation of and Experiments with ABML and MLBA, Aspic deliverable d3.4.

Možina, M., Zabkar, J. & Bratko, I. (2007), 'Argument Based Machine Learning', *Artificial Intelligence* **171**(10-15), 922–937.

Napierala, K. & Stefanowski, J. (2010), Argument Based Generalization of MODLEM Rule Induction Algorithm, *in* M. Szczuka, M. Kryszkiewicz, S. Ramanna, R. Jensen & Q. Hu, eds, 'Rough Sets and Current Trends in Computing', Vol. 6086 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 138–147.

National Highway Traffic Safety Administration (2012), Fatality Analysis Reporting System (FARS) - Analytical Users Manual 1975-2011.

National Highway Traffic Safety Administration (2016), 'Fatality Analysis Reporting System (FARS) - Detailing the Factors behin Traffic Fatalities on our Roads'.
**URL:** *http://www.nhtsa.gov/FARS (last visited: 2016-09-26)*

Nehaniv, C. L. & Dauntenham, K., eds (2007), *Imitation and Social Learning in Robots, Humans and Animals: Behavioral, Social and Communicative Dimensions*, Cambridge University Press, Cambridge, United Kingdom.

Nielsen, S. (2006), Reasoning and Decision Making with Multiple Autonomous Agents, Doctoral dissertation, Aalborg University.

Nielsen, S. & Parsons, S. (2007), 'An Application of Formal Argumentation: Fusing Bayesian Networks in Multi-agent Systems', *Artificial Intelligence* **171**(10-15), 754–775.

Nielsen, U., Pellet, J.-P. & Elisseeff, A. (2008), Explanation Trees for Causal Bayesian Networks, *in* D. McAllester & P. Myllymaki, eds, 'Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI-08)', AUAI Press, Corvallis, Oregon, pp. 427–434.

Nikolai, C. & Madey, G. (2009), 'Tools of the Trade: A Survey of Various Agent Based Modeling Platforms', *Journal of Artificial Societies and Social Simulation* **12**(2), 2.

Noble, J. & Franks, D. W. (2003), 'Social Learning in a Multi-Agent System.', *Computing and Informatics* **22**(6), 1001–1015.

Noble, J. & Todd, P. M. (2002), Imitation or Something Simpler? Modeling Simple Mechanisms for Social Information Processing, Imitation in Animals and Artifacts, *in* K. Dautenhahn & C. L. Nehaniv, eds, 'Imitation in Animals and Artifacts', MIT Press, Cambridge, MA, Cambidge, MA, USA, pp. 423–439.

Nonaka, I. (1994), 'A Dynamic Theory of Organizational Knowledge Creation', *Organization science* **5**(1), 14–37.

Nonaka, I. & Takeuchi, H. (1995), *The knowledge-creating company: How Japanese companies create the dynamics of innovation*, Oxford University Press, USA.

Nonaka, I., Von Krogh, G. & Voelpel, S. (2006), 'Organizational Knowledge Creation Theory: Evolutionary Paths and Future Advances', *Organization studies* **27**(8), 1179–1208.

Nunes, L. & Oliveira, E. (2003), Cooperative Learning Using Advice Exchange, *in* E. Alonso, D. Kudenko & D. Kazakov, eds, 'Adaptive Agents and Multi-Agent Systems', Vol. 2636 of *Lecture Notes in Computer Science*, Springer, pp. 33–48.

Nunes, L. & Oliveira, E. (2004), Learning from Multiple Sources, *in* 'Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 3', AAMAS '04, IEEE Computer Society, Washington, DC, USA, pp. 1106–1113.

O'Brien, P. D. & Nicol, R. C. (1998), 'FIPA—Towards a Standard for Software Agents', *BT Technology Journal* **16**(3), 51–59.

Ontañón, S., Dellunde, P., Godo, L. & Plaza, E. (2010), Towards a Logical Model of Induction from Examples and Communication, *in* R. Alquézar, A. Moreno & J. Aguilar, eds, 'Artificial Intelligence Research and Development - Proceedings of the 13th International Conference of the Catalan Association for Artificial Intelligence', IOS Press, L'Espluga de Francolí, Spain, pp. 259 – 268.

Ontañón, S. & Plaza, E. (2007*a*), Case-based Learning from Proactive Communication, *in* R. Sangal, H. Mehta & R. K. Bagga, eds, 'Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI '07)', Morgan Kaufmann Publishers Inc., Hyderabad, India, pp. 999–1004.

Ontañón, S. & Plaza, E. (2007*b*), Learning and Joint Deliberation through Argumentation in Multiagent Systems, *in* 'Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '07)', ACM Press, New York, New York, USA, pp. 975–982.

Ontañón, S. & Plaza, E. (2009), Argumentation-based Distributed Induction, *in* M. Esteva, A. Fernandez & A. Giret, eds, 'Proceedings of the 2nd Workshop on Agreement Technologies 2009 (in CAEPIA 2009)', Sevilla, Spain, pp. 154–168.

Ontañón, S. & Plaza, E. (2010*a*), Argumentation-based Example Interchange for Multiagent Induction, *in* 'Proceedings of the 2010 Conference on Artificial Intelligence Research and Development: Proceedings of the 13th International Conference of the Catalan Association for Artificial Intelligence', IOS Press, Amsterdam, The Netherlands, pp. 59–68.

Ontañón, S. & Plaza, E. (2010*b*), Concept Convergence in Empirical Domains, *in* B. Pfahringer, G. Holmes & A. Hoffmann, eds, 'Discovery Science: 13th International Conference, DS 2010, Canberra, Australia, October 6-8, 2010. Proceedings', Springer Berlin Heidelberg, pp. 281 – 295.

Ontañón, S. & Plaza, E. (2010*c*), Empirical Argumentation: Integrating Induction and Argumentation in MAS, *in* P. McBurney, I. Rahwan & S. Parsons, eds, 'Argumentation in Multi-Agent Systems: 7th International Workshop, ArgMAS 2010 Toronto, ON, Canada, May 10, 2010 Revised, Selected and Invited Papers', Springer Berlin Heidelberg, pp. 163–180.

Ontañón, S. & Plaza, E. (2010*d*), Multiagent Inductive Learning: An Argumentation-based Approach, *in* J. Fürnkranz & T. Joachims, eds, 'Proceedings of the 27th International Conference on Machine Learning', number 2, Omnipress, Haifa, Israel, pp. 839 – 846.

Open Refine (2016), 'Open Refine - A Free, Open Source, Powerful Tool for Working with Messy Data'.
**URL:** *http://openrefine.org/ (last visited: 2016-09-27)*

Oquab, M., Bottou, L., Laptev, I. & Sivic, J. (2014), Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks, *in* '2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)', Vol. 00, pp. 1717–1724.
**URL:** *doi.ieeecomputersociety.org/10.1109/CVPR.2014.222*

Pan, S. J. & Yang, Q. (2010), 'A Survey on Transfer Learning', *Knowledge and Data Engineering, IEEE Transactions on* **22**(10), 1345–1359.

Panait, L. & Luke, S. (2005), 'Cooperative Multi-Agent Learning: The State of the Art', *Autonomous Agents and Multi-Agent Systems* **11**(3), 387–434.

Pandas (2016), 'pandas - Python Data Analysis Library'.
**URL:** *http://pandas.pydata.org/ (last visited: 2016-09-27)*

Pantke, F. (2017), Kennzahlenbasierte Steuerung, Koordination und Aktionsplanung in Multiagentensystemen, Doctoral dissertation, Universität Bremen.

Pawlaszczyk, D. (2009), Skalierbare agentenbasierte Simulation: Werkzeuge und Techniken zur Verteilten Ausführung agentenbasierter Modelle, PhD thesis, Universität Ilmenau.

Penrose, E. T. (1995), *The Theory of the Growth of the Firm*, Oxford University Press, USA.

Pokahr, A., Braubach, L. & Lamersdorf, W. (2005), Jadex: A BDI reasoning engine, *in* R. H. Bordini, M. Dastani, J. Dix & A. E. Seghrouchini, eds, 'Multi-Agent Programming: Languages, Platforms and Applications', Springer Berlin / Heidelberg, pp. 149–174.

Porzel, R. & Warden, T. (2010), Working Simulations with a Foundational Ontology, *in* K. Schill, B. Scholz-Reiter & L. Frommberger, eds, 'Artificial Intelligence and Logistics (AiLog) - Workshop at ECAI 2010', Lisbon, Portugal, pp. 67–72.

Poslad, S. (2007), 'Specifying Protocols for Multi-agent Systems Interaction', *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* **2**(4), 15.

Poslad, S., Buckle, P. & Hadingham, R. (2000), The FIPA-OS Agent Platform: Open Source for Open Standards, *in* 'Proceedings of the 5th International Conference and Exhibition on the practical Application of Intelligent Agents and Multi-agents', Manchester, UK, pp. 355–368.

Preece, A., Hui, K. Y., Gray, A., Marti, P., Bench-Capon, T., Cui, Z. & Jones, D. (2001), 'KRAFT: An Agent Architecture for Knowledge Fusion', *International Journal of Cooperative Information Systems* **10**(1-2), 171–196.

Preece, A., Hui, K. Y., Gray, A., Marti, P., Bench-Capon, T., Jones, D. & Cui, Z. (2000), 'The KRAFT Architecture for Knowledge Fusion and Transformation', *Knowledge-Based Systems* **13**(2), 113–120.

Probst, G., Raub, S. & Rombardt, K. (2006), *Wissen managen - Wie Unternehmen ihre wertvollste Ressource nutzen*, 5 edn, Gabler.

Quinlan, J. R. (1987*a*), Generating Production Rules from Decision Trees, *in* 'Proceedings of the 10th International Joint Conference on Artificial Intelligence', Vol. 1, Morgan Kaufmann Publishers Inc., Milan, Italy, pp. 304–307.

Quinlan, J. R. (1987*b*), 'Simplifying Decision Trees', *International Journal of Man-Machine Studies* **27**(3), 221–234.

Quinlan, J. R. (1993), *C4. 5: Programs for Machine Learning*, Morgan Kaufmann.

Raghavan, H., Madani, O. & Jones, R. (2006), 'Active Learning with Feedback on Features and Instances', *The Journal of Machine Learning Research* **7**, 1655–1686.

Rao, A. S. & Georgeff, M. P. (1997), Modeling Rational Agents within a BDI-Architecture, *in* M. N. Huhns & M. P. Singh, eds, 'Readings in agents', Morgan Kaufmann Publishers Inc., San Francisco, CA, pp. 317–328.

Rhodes, B. J. & Maes, P. (2000), 'Just-in-time Information Retrieval Agents', *IBM Systems Journal* **39**(3.4), 685–704.

Roberts, C. A. & Dessouky, Y. M. (1998), 'An overview of object-oriented simulation', *Simulation* **70**(6), 359–368.

Rojas, R. (2013), *Neural networks: a systematic introduction*, Springer Science & Business Media.

Russell, S. J. & Norvig, P. (2010), *Artificial Intelligence: A Modern Approach*, 3 edn, Prentice Hall.

Sato, G., Azevedo, H. & Barthès, J.-P. (2012), 'Agent and Multi-agent Applications to Support Distributed Communities of Practice: A Short Review', *Autonomous Agents and Multi-Agent Systems* **25**(1), 87–129.

Scholz-Reiter, B., de Beer, C., Freitag, M. & Jagalski, T. (2008), 'Bio-inspired and Pheromone-based Shop-floor Control', *International Journal of Computer Integrated Manufacturing* **21**(2), 201–205.

Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., de Velde, W. & Wielinga, B. (1999), *Knowledge Engineering and Management: The CommonKADS Methodology*, MIT press.

Schubert, P., Lincke, D. & Schmid, B. (1998), A Global Knowledge Medium as a Virtual Community: The NetAcademy Concept, *in* E. Hoadley & I. Benbast, eds, 'Proceedings of the 4th Americas Conference on Information Systems', Baltimore, MD, USA, pp. 618–620.

Schuldt, A. (2009), Decentralisation and Interaction Efficiency in Cooperating Autonomous Logistics Processes, *in* H.-J. Kreowski, B. Scholz-Reiter & K.-D. Thoben, eds, '2nd International Conference on Dynamics in Logistics (LDIC 2009)', Springer Berlin Heidelberg, Bremen, Germany, pp. 269–278.

Schuldt, A. (2011), *Multiagent Coordination Enabling Autonomous Logistics*, Springer-Verlag, Heidelberg, Germany.

Sen, S. & Kar, P. P. (2002), 'Sharing a Concept', Working Notes of the AAAI Spring Symposium on Collaborative Learning Agents.

Settles, B. (2009), Active Learning Literature Survey, Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

Shon, A. P., Verma, D. & Rao, R. P. N. (2007), Active Imitation Learning, *in* A. Cohn, ed., 'Proceedings of the 22nd National Conference on Artificial intelligence (AAAI'07)', Vol. 1, AAAI Press, Vancouver, Brithish Columbia, Canada, pp. 756–762.

Sindhwani, V., Melville, P. & Lawrence, R. D. (2009), Uncertainty Sampling and Transductive Experimental Design for Active Dual Supervision, *in* 'Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09', ACM Press, New York, New York, USA, pp. 1–8.

Singh, M. P. (2003), Agent Communication Languages: Rethinking the Principles, *in* M.-P. Huget, ed., 'Communication in Multiagent Systems: Agent Communication Languages and Conversation Policies', Springer Berlin Heidelberg, pp. 37–50.

Smirnov, A., Pashkin, M., Chilov, N. & Levashova, T. (2002), Multi-agent Architecture for Knowledge Fusion from Distributed Sources, *in* B. Dunin-Keplicz & E. Nawarecki, eds, 'From Theory to Practice in Multi-Agent Systems: Second International Workshop of Central and Eastern Europe on Multi-Agent Systems, CEEMAS 2001 Cracow, Poland, September 26–29, 2001 Revised Papers', Springer Berlin Heidelberg, pp. 293–302.

Smith, R. G. (1977), The Contract Net: A Formalism for the Control of Distributed Problem Solving, *in* R. Reddy, ed., '5th International Joint Conference on Artificial Intelligence (IJCAI 1977)', William Kaufmann, Cambridge, MA, USA, p. 472.

Sokolova, M. & Lapalme, G. (2009), 'A Systematic Analysis of Performance Measures for Classification Tasks', *Information Processing & Management* **45**(4), 427–437.

Staab, S. & Schnurr, H. P. (2000), 'Smart Task Support through Proactive Access to Organizational Memory', *Knowledge-based Systems* **13**(5), 251–260.

Steels, L. (2003), 'Evolving Grounded Communication for Robots', *Trends in Cognitive Sciences* **7**(7), 308–312.

Stefanowski, J. (1998), The Rough Set Based Rule Induction Technique for Classification Problems, *in* 'Proceedings of 6th European Conference on Intelligent Techniques and Soft Computing', Aachen, Germany, pp. 109–113.

Stone, P. & Veloso, M. (2000), 'Multiagent Systems: A Survey from a Machine Learning Perspective', *Autonomous Robots* **8**(3), 345–383.

Timm, I. J. (2003), Dynamisches Konfliktmanagement als Verhaltenssteuerung intelligenter Agenten, PhD thesis, Universität Bremen, Bremen, Germany.

Timm, I. J., Scholz, T., Herzog, O., Krempels, K.-H. & Spaniol, O. (2006), From agents to multiagent systems, *in* 'Multiagent Engineering', Springer, pp. 35–51.

Tomanek, K. & Hahn, U. (2009), Semi-supervised Active Learning for Sequence Labeling, *in* 'Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2', Suntec, Singapore, pp. 1039–1047.

Tomasello, M. (1999), *The Cultural Origins of Human Cognition*, Havard University Press, Cambridge, MA, USA.

Tomasello, M., Kruger, A. C. & Ratner, H. H. (1993), 'Cultural Learning', *Behavioural and Brain Sciences* (16), 495–552.

Tožička, J., Jakob, M. & Pěchouček, M. (2006), Market-Inspired Approach to Collaborative Learning, *in* M. Klusch, M. Rovatsos & T. R. Payne, eds, 'Proceedings of the 10th International Conference on Cooperative Information Agents', Springer Berlin Heidelberg, Edinburgh, UK, pp. 213–227.

Tuomi, I. (1999), Data is More than Knowledge: Implications of the Reversed Knowledge Hierarchy for Knowledge Management and Organizational Memory, *in* 'Journal of Management Information Systems', Vol. 16, IEEE, pp. 103–117.

Uckelmann, D., Harrison, M. & Michahelles, F. (2011), *Architecting the Internet of Things*, Springer-Verlag New York Inc.

Vahrenkamp, R. (2007), *Logistik: Management und Strategien*, 6 edn, Oldenbourg Wissenschaftsverlag.

van Diggelen, J., Dignum, V., van Elst, L. & Abecker, A., eds (2005), *Proceedings of the AAMAS 2005 Workshop on Agent Mediated Knowledge Management*, Utrecht, The Netherlands.

Van Dyke Parunak, H., Savit, R. & Riolo, R. (1998), Agent-based Modeling vs. Equation-based Modeling: A Case Study and Users' Guide, *in* J. S. Sichman, R. Conte & N. Gilbert, eds, 'Multi-Agent Systems and Agent-Based Simulation', Springer Berlin / Heidelberg, pp. 277–283.

van Elst, L., Dignum, V. & Abecker, A. (2004), Towards Agent-Mediated Knowledge Management, *in* L. van Elst, V. Dignum & A. Abecker, eds, 'Agent-Mediated Knowledge Management: International Symposium AMKM 2003, Stanford, CA, USA, March 24-26, 2003, Revised and Invited Papers', Springer Berlin Heidelberg, pp. 1–30.

Van Leeuwen, E. H. & Norrie, D. (1997), 'Holons and Holarchies', *Manufacturing Engineer* **76**(2), 86–88.

Visser, U. & Burkhard, H.-D. (2007), 'RoboCup: 10 Years of Achievements and Future Challenges', *AI magazine* **28**(2), 115.

Von Krogh, G. (1998), 'Care in Knowledge Creation', *California Management Review* **40**(3), 133–153.

Warden, T. & Herzog, O. (2012), Organizing Goal-oriented Knowledge Exchange in Multiagent Systems, *in* '1st Joint International Symposium on System-Integrated Intelligence 2012. New Challenges for Product and Production Engineering'.

Warden, T., Porzel, R., Gehrke, J. D., Herzog, O., Langer, H. & Malaka, R. (2010), Towards Ontology-based Multiagent Simulations: The PlaSMA Approach, *in* A. Bargiela, S. Azam Ali, D. Crowley & E. J. H. Kerckhoffs, eds, '24th European Conference on Modelling and Simulation (ECMS '10)', European Council for Modelling and Simulation, Kuala Lumpur, KL, Malaysia, pp. 50–56.

Warden, T., Porzel, R., Gehrke, J. D., Langer, H., Herzog, O. & Malaka, R. (2011), Knowledge Management for Agent-based Control under Temporal Bounds, *in* M. Hülsmann, B. Scholz-Reiter & K. Windt, eds, 'Autonomous Cooperation and Control in Logistics: Contributions and Limitations - Theoretical and Practical Perspectives', Springer Berlin Heidelberg, pp. 229–246.

Warden, T. & Visser, U. (2012), 'Real-time Spatio-temporal Analysis of Dynamic Scenes', *Knowledge and Information Systems* **32**(2), 243–279.

Warden, T. & Wojtusiak, J. (2010), Learnable Evolutionary Optimization in Autonomous Pickup & Delivery Planning, Technical report, Center for Computing and Communication Technologies, Bremen.

Warden, T., Wojtusiak, J. & Herzog, O. (2012), Intelligent Modelling and Control for Autonomous Logistics, *in* J. Kołodziej, S. U. Khan & T. Burczyński, eds, 'Advances in Intelligent Modelling and Simulation: Artificial Intelligence-based Models and Techniques in Scalable Computing', Springer Berlin Heidelberg, pp. 295–325.

Weiß, G. & Dillenbourg, P. (1999), What is Multi in Multiagent Learning, *in* P. Dillenbourg, ed., 'Collaborative Learning. Cognitive and Computational Approaches', 1 edn, Emerald Group Publishing Limited, pp. 64–80.

Weiß, G. & Jakob, R. (2005), *Agentenorientierte Softwareentwicklung: Methoden und Tools*, Springer Berlin Heidelberg.

Wiederhold, G. & Genesereth, M. (1997), 'The conceptual Basis for Mediation Services', *IEEE Expert* **12**(5), 38–47.

Windt, K., Jeken, O. & Arbabzadah, A. (2010), Improved Logistics Performance through the Use of Locked Flexibility Potentials, *in* W. Sihn & P. Kuhlang, eds, 'Proceedings of the 43rd CIRP International Conference on Manufacturing Systems. Sustainable Production and Logistics in Global Networks', NW Verlag, Vienna, Vienna, Austria, pp. 892–899.

Winikoff, M. (2005), Jack$^{TM}$ Intelligent Agents: An Industrial Strength Platform, *in* R. H. Bordini, M. Dastani, J. Dix & A. El Fallah Seghrouchni, eds, 'Multi-Agent Programming: Languages, Platforms and Applications', Springer US, Boston, MA, pp. 175–193.

Witten, I. H., Frank, E. & Hall, M. A. (2011), *Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques*, 3. edn, Morgan Kaufmann.

Wojtusiak, J., Michalski, R., Kaufman, K. & Pietrzykowski, J. (2006), The AQ21 Natural Induction Program for Pattern Discovery: Initial Version and its Novel Features, *in* '2006 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06)', IEEE, Arlington, VA, USA, pp. 523–526.

Wojtusiak, J., Warden, T. & Herzog, O. (2011), Agent-based Pickup and Delivery Planning: The Learnable Evolution Model Approach, *in* '2011 International Conference on Complex, Intelligent, and Software Intensive Systems', Seoul, South Korea, pp. 1–8.

Wojtusiak, J., Warden, T. & Herzog, O. (2012*a*), 'Machine Learning in Agent-based Stochastic Simulation: Inferential Theory and Evaluation in Transportation Logistics', *Computers and Mathematics with Applications* **64**(12), 3658—-3665.

Wojtusiak, J., Warden, T. & Herzog, O. (2012*b*), 'The Learnable Evolution Model in Agent-based Delivery Optimization', *Memetic Computing* **4**(3), 165–181.

Wooldridge, M. (2002), *An Introduction to Multiagent Systems*, John Wiley & Sons, Chichester, UK.

Wooldridge, M. & Jennings, N. R. (1995), 'Intelligent Agents: Theory and Practice', *Knowledge Engineering Review* **19**(2), 115–152.

Wooldridge, M. & Jennings, N. R. (1999), 'The Cooperative Problem-solving Process', *Journal of Logic and Computation* **9**(4), 563–592.

Wurst, M. (2005), Evaluating Knowledge Management in Heterogeneous Domains by Agent-based Simulation, *in* J. van Diggelen, V. Dignum, L. van Elst & A. Abecker, eds, 'Proceedings of the AAMAS 2005 Workshop on Agent Mediated Knowledge Management', Utrecht, The Netherlands, pp. 82–96.

Xing, X., Warden, T., Nicolai, T. & Herzog, O. (2009), SMIZE: A Spontaneous Ride-Sharing System for Individual Urban Transit, *in* L. Braubach, W. van der Hoek, P. Petta & A. Pokhar, eds, 'Multiagent System Technologies: 7th German Conference, MATES 2009, Hamburg, Germany, September 9-11, 2009. Proceedings', Springer Berlin Heidelberg, pp. 165–176.

Xu, J., Yao, L. & Li, L. (2015), 'Argumentation Based Joint Learning: A Novel Ensemble Learning Approach', *PLOS ONE* **10**(5), 1–21.

Yarowsky, D. (1995), Unsupervised Word Sense Disambiguation Rivaling Supervised Methods, *in* 'Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics', Association for Computational Linguistics, Cambidge, MA, USA, pp. 189–196.

Yu, B., Venkatraman, M. & Singh, M. P. (2003), 'An Adaptive Social Network for Information Access: Theoretical and Experimental Results', *Applied Artificial Intelligence* **17**(1), 21–38.

Zambonelli, F., Jennings, N. R. & Wooldridge, M. (2003), 'Developing Multiagent Systems: The Gaia Methodology', *ACM Transactions on Software Engineering and Methodology (TOSEM)* **12**(3), 317–370.

Zhang, Y., Su, H., Jia, T. & Chu, J. (2005), Rule Extraction from Trained Support Vector Machines, *in* T. Ho, D. Cheung & H. Liu, eds, 'Advances in Knowledge Discovery and Data Mining: 9th Pacific-Asia Conference, PAKDD 2005, Hanoi, Vietnam, May 18-20, 2005. Proceedings', Springer Berlin Heidelberg, pp. 92–95.

Zhou, Y. & Goldman, S. (2004), Democratic Co-learning, *in* 'Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence', IEEE Computer Society, Boca Raton, FL, USA, pp. 594–602.

Zhou, Z.-H. & Li, M. (2005), 'Tri-Training: Exploiting Unlabeled Data Using Three Classifiers', *IEEE Transactions on Knowledge and Data Engineering* **17**(11), 1529–1541.

Zhu, X. (2005), Semi-supervised Learning Literature Survey, Computer sciences technical report 1530, University of Madison-Wisconsin.

Zilke, J. R., Loza Mencía, E. & Janssen, F. (2016), DeepRED – Rule Extraction from Deep Neural Networks, *in* T. Calders, M. Ceci & D. Malerba, eds, 'Discovery Science', Springer International Publishing, Cham, pp. 457–473.

Zliobaite, I. (2009), Learning under Concept Drift: An Overview, Technical report, Vilnius University, Faculty of Mathematics and Informatics.

# Appendices

Appendix A

# Chapter 5 Supplements

On several occasions, Chapter 5 referenced techniques and formalisations that constitute enabling contributions by third parties. Consequently, these are presented here in the appendix so as not to allow for confusion with the original work presented in Chapter 5.

## A.1 Basic Notions on Clustering

Let $B$ denote a set of objects that are to be clustered. An object $\mathbf{b}_i \in B$ is a *core object*, iff $|N_\epsilon(\mathbf{b}_i)| \geq t_{min} \;:\; N_\epsilon(\mathbf{b}) := \{\mathbf{b} \in B \mid dist(\mathbf{b}_i, \mathbf{b}) \leq \epsilon\}$. For such an object it holds that within its epsilon neighbourhood $N_\epsilon(\mathbf{b}_i)$ reside at least $t_{min} \in \mathbb{N}_0^+$ further objects. The parameter tuple $\langle\, \epsilon, t_{min} \,\rangle$ defines a desired density threshold.

An object $\mathbf{b}_j \in B$ is *directly density-reachable* from $\mathbf{b}_i \in B$ with regard to $\langle\, \epsilon, t_{min} \,\rangle$ iff $\mathbf{b}_j \in N_\epsilon(\mathbf{b}_i)$ and $\mathbf{b}_i$ is a core object in $B$ (see Figure A.1).

An object $\mathbf{b}_i \in B$ is *density-reachable* from $\mathbf{b}_i \in B$ with regard to $\langle\, \epsilon, t_{min} \,\rangle$ iff there exists a sequence of objects $seq(\mathbf{b}_1, \ldots, \mathbf{b}_n) \;:\; \mathbf{b}_i = \mathbf{b}_1, \mathbf{b}_j = \mathbf{b}_n$ and $\mathbf{b}_{k+1}$ is directly density-reachable from $\mathbf{b}_k$ with respect to $\langle\, \epsilon, t_{min} \,\rangle$ for all $1 \leq k \leq n$.

Also, an object $\mathbf{b}_j \in B$ is *density-connected* with another object $\mathbf{b}_i \in B$ with regards to $\langle\, \epsilon, t_{min} \,\rangle$ iff an object $\mathbf{b}_k \in B$ exists such that both initially noted points a density-reachable from the latter object (see Figure A.1).

Based on the basic notions presented above, a *density-based cluster* with respect to $\langle\, \epsilon, t_{min} \,\rangle$ in $B$ is a non-empty subset of the data where all included objects are density-connected and where all objects that are density-reachable from any core object also belong to the cluster. Such clusters have an important property which is utilised in the DBSCAN algorithm.

**Lemma A.1.** *Let a density-based cluster with respect to* $\langle\, \epsilon, t_{min} \,\rangle$ *in a data set* $B$ *be denoted as* $Cl$ *and let further* $\mathbf{b}_c \in Cl$ *be a core object. Then it holds that* $Cl := \{\mathbf{b} \in B \mid \mathbf{b}$ *is density-reachable from* $\mathbf{b}_c\}$.

A *density-based clustering* as computed by DBSCAN corresponds to the complete

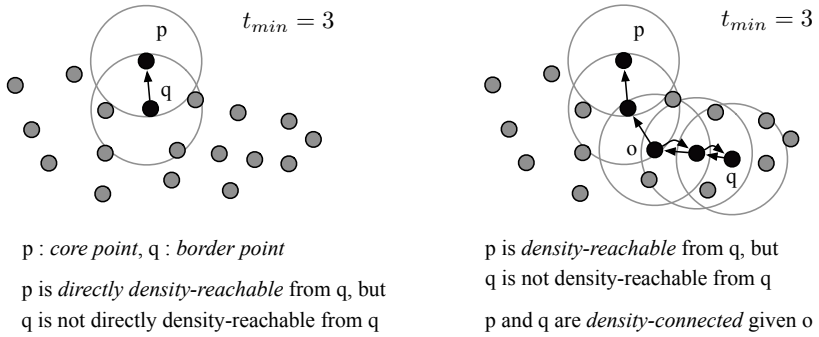



**Figure A.1:** Basic notions for density-based clustering, adapted from (Ester et al. 1996).

set of clusters $Cl_i$ in a data set with respect to $\langle \epsilon, t_{min} \rangle$. Those objects which cannot be associated to any cluster are understood as noise.

As highlighted in (Ester & Sander 2000, p. 71), density-based clusters as defined above can overlap only for such non-core objects. However, such an overlap may occur only for parameterisations where $t_{min} \geq 4$ according to the following lemma:

**Lemma A.2.** *Let Clusters denote a clustering of the data set B with respect to* $\langle \epsilon, minPoints \rangle$. *For all* $Cl_i, Cl_j \in Clusters$ *it holds that: a) If* $Cl_i \neq Cl_j$, *then for all* $\mathbf{b} \in Cl_i \cap Cl_j$, $|N_\epsilon(\mathbf{b})| < t_{min}$ *such that* $\mathbf{b}$ *is no core object. b) If* $t_{min} \leq 3$ *and* $Cl_i \neq Cl_j$, *then* $Cl_i \cap Cl_j = \emptyset$, *such that Clusters is free of overlaps.*

## A.2 Argument Based Machine Learning using ABCN2

As shown in Figure A.2, argument advice which is procured for a specific critical instance constitutes instance-specific background knowledge designed to constrain those rules induced by the ABCN2 learner, which cover the respective instance. In CN2, induced rules have the general form

**If** *Complex* **Then** $C = y$

The *complex*, which is also referred to as *premise* of the rule, is a logical conjunction of simple conditions which are called *selectors*. For nominal features, a selector specifies a value from the feature domain, such as *color* = BLACK. For numeric features, the selector typically specifies a value range by means of a threshold value, for instance $size > 120.0$ [cm]. If the premise of a rule is satisfied by the feature values of an instance $\langle \mathbf{x}, y \rangle$, then the rule *covers* the instance.

**Definition A.1** (AB-consistency)

*A rule, denoted here as ru, is said to be consistent with an argument* $arg \in Arg^+ \cup Arg^-$ *if for all reasons, it holds that*

*a) If the reason* $r_i$ *is in one of the forms a)* $r_i = \langle F_i, =, v \rangle : v \in Dom(F_i)$, *or b)* $r_i = \langle F_i, \{>|\geq\}, \{t \mid *\} \rangle$, *or c)* $r_i = \langle F_i, \{<|\leq\}, \{t \mid *\} \rangle$, *then* $r_i$

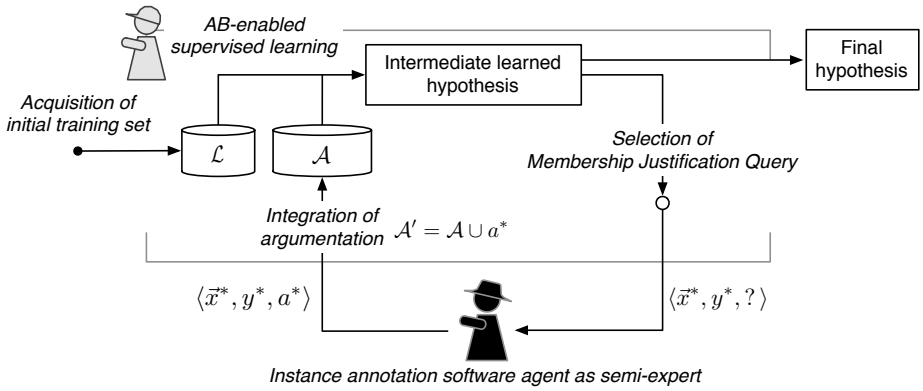



**Figure A.2:** The active learning cycle for argument based machine learning applied in a multiagent environment, assuming a 1:1 interaction pattern (see Section 5.4.2).

*needs to appear as a selector in the premise of ru.*

*b) If the reason $r_i$ is present in an underspecified form where no explicit threshold value is given (as in $r_i = \langle F_i, >, * \rangle$), then the premise of ru needs to contain a selector which is a concrete specialisation of $r_i$. I.e., the selector does specify a concrete threshold value. However, it is not of importance with respect to consistency.*

When argument-annotated instances are used in ABCN2, the definition of rule coverage with respect to an instance needs to be refined as follows.

**Definition A.2** (AB-coverage)

*Let an instance with associated arguments be denoted as $ae = \langle \mathbf{x}, y, args^+, args^- \rangle$ where $args^+ \subset Args^+ \setminus \emptyset$, $args^- \subset Args^- \cup \emptyset$ (see Definitions 5.13 and 5.14). Further assume a propositional rule ru of the form 'If complex Then $C = y$'. Then, said rule* AB-*covers the instance, written as* AB-COVERS$(ru, ae)$*, if*

*a)* $\forall\, sel \in ru.complex$ : HOLDS$(ae, sel)$ = TRUE *(same as in CN2)*

*b)* $\exists\, arg \in args^+$ : AB-CONSISTENT$(ru, arg)$

*c)* $\forall\, arg \in args^-$ : $\neg$AB-CONSISTENT$(ru, arg)$

### The ABCN2 Algorithm

The CN2 algorithm, which has acted as the point of origin for the development of the ABCN2 version, consists of a covering algorithm and a search procedure that finds individual rules by means of a local beam search. The covering algorithm controls the complete induction process. It seeks to cover all instances which are procured in the respective training set. In its unordered variant, the covering algorithm simply iterates over all possible classes of the target concept and calls a sub-procedure CN2FORONECLASS() which starts by the search for a rule which covers some instances of the current class. It then removes exactly these instances, adds the induced rule to the rule set and begins another iteration. This process is repeated until all instances of the class are covered.

**Algorithm 6** Covering algorithm of ABCN2 algorithm that learns rules from instances for given class. Algorithm adopted from (Možina et al. 2007, p. 927)

**Input:** $ES$ : The set of instances used as learning input, given as $\langle \mathbf{x}, y, arguments \rangle$ where $arguments$ is a set of arguments, either empty or containing at least one inclusive argument and optionally exclusive arguments
$AES$ : List of argument-annotated instances, initially empty
$y \in \mathcal{Y}$ : A class of the target concept
**Output:** $RuleList$ : List of learned rules, initially the empty set

1: **function** ABCN2ForOneClass($ES, y$)
2: $AES \leftarrow \{ae \in ES \mid ae.arguments \neq \emptyset\}$, hence $AES \subseteq ES$
3: **Find splits** for arguments where threshold not specified
4: **Evaluate** arguments (as if they were rules) of instances in $AES$ and sort examples in $AES$ according to the evaluations of their best argument.
5: **while** $AES \neq \emptyset$ **do**
6: $AE \leftarrow$ chooseFirst($AES$)
7: $BestRule \leftarrow$ ABFindBestRule($ES, AE, y$)
8: $RuleList \leftarrow RuleList \cup BestRule$
9: $AES \leftarrow \{ae \in AES \mid \neg$isABCovered$(ae, BestRule)\}$
10: **for** $Rule \in RuleList$ **do**
11: $ES \leftarrow \{ex \in ES \mid \neg$isABCovered$(ex, Rule)\}$
12: $CN2Rules \leftarrow$ CN2ForOneClass($ES, y$) `% Use CN2 to cover remaining inst.`
13: **return** $RuleList \cup CN2Rules$

Argument based machine learning requires that an induced model actually explains argument-annotated instances by means of the procured pool of arguments. For rule learning, this means that for each argument-annotated instance, the target rule set must contain at least one rule that AB-covers this instance according to the definition provided above.

Možina and colleagues have proposed to ensure this requirement via a modification of the CN2ForOneClass() procedure. Specifically, they substitute the traditional CN2 covering semantics with AB-covering as illustrated in Algorithm 6. The resulting procedure ABCN2ForOneClass() is designed such that it prefers explaining as many as possible non-annotated instances by arguments given for the few argument-annotated instances. The procedure first retrieves the list of training instances for which an annotation with arguments, i.e., an argumentation, has already been contributed by an advisor. Subsequently, any arguments that are contained within this set which have underspecified selectors are completed. Specifically, proper concrete thresholds are determined that are in line with the advisee training instances. In the following step, the list of argument-annotated instances is sorted according to an evaluation of the respective arguments. For this evaluation, the arguments are treated like proper rules of the form "If $argument$ Then $class = y$" and evaluated with the same evaluation function. At the end of this operation, the annotated instances are

ordered according to the relevance of their respective best argument.

While the list of annotated instances is not empty, the first (best remaining) instance is chosen and the covering algorithm calls the sub procedure ABCN2ForOneClass() shown in Algorithm 7. This procedure accepts training instances, a single annotated instance and a target class. It performs a beam search for the best rule, which AB-covers the annotated instance. Its result is added to the complete rule set induced thus far for the current class and all annotated instances which are already AB-covered, according to Definition A.2 by the new rule are removed from the sorted list of annotated instances that still need to be processed. The removal of all positive examples is not necessary, as each of the argument-annotated instances differently constrains the search and thus prevents ABCN2 from inducing the same rule again.

When no more annotated instances are left for processing, the full set of training instances is filtered such that all instances that are AB-covered by the hitherto induced rules are removed. Since the remaining instance are not covered with respect to the expert-provided arguments the regular CN2 covering algorithm CN2ForOneClass() (see (Clark & Boswell 1991, Clark & Niblett 1989)) is applied to induce additional rules.

The beam search which is shown in Algorithm 7 is an adapted version of the regular beam search procedure found in CN2 implementations. It features several modifications which taken together make sure that it finds a rule which is guaranteed to AB-cover the argument-annotated instance $AE$ which is presented as an additional parameter. In contrast to the classical CN2, the Star is not initially assigned with the empty complex. Instead, it is initialised with complexes that correspond to the inclusive arguments for the classification of $AE$. The rationale here is that the sought rule needs to AB-cover $AE$ an thus needs to contain the reasons of at least one of the inclusive arguments. This is best guaranteed when the process of rule specialisation begins with the aforementioned complexes.

When complexes that are currently stored within the Star are specialised by the addition of additional selectors, these selectors must satisfy the condition that they hold true for $AE$. This ensured the coverage of the seed instance by the induced rule. Another means to retain the desired property of AB-coverage is applied. The set of all extended complexes is post-processed such that all complexes are removed that are consistent with an exclusive argument specified for $AE$.

Finally, rules that contain only conditions that that are present in inclusive arguments are likely to be consistent with domain knowledge. This is why by means of $STAR^{AB}_{new}$, Algorithm 7 retains the most promising of these rules, even if they are not currently among the n best rule candidates with respect to the applied rule quality measure.

**Algorithm 7** Algorithm that finds best rule that AB-covers argument-annotated instance. The "quality" of a complex is evaluated by user-defined evaluation function. Algorithm adopted from (Možina et al. 2007, p. 928)

**Input:** $ES$ : The set of instances used as learning input
$y \in \mathcal{Y}$ : A class of the target concept
**Output:** *RuleList* : A list of learned rules, initially the empty set

**function** ABFindBestRule($ES, AE, y$)
  **let** the set $STAR$ contain positive arguments of $AE$ (written as complexes)
  **Evaluate** complexes in $STAR$ (using quality function).
  **let** $C_{best}$ be the best complex from $STAR$.
  **let** $SELECTORS$ be the set of all possible selectors that are true for $AE$.
  **let** ARG_REASONS be the set of all reasons in positive arguments of AE.
  **while** $STAR \neq \emptyset$ **do**
    { Specialise all complexes in $STAR$ as follows }
    **let** $STAR_{new}$ be the set $\{x \wedge y \mid x \in STAR, y \in SELECTORS\}$
    **Remove** from $STAR_{new}$ all complexes consistent with any negative argument of $AE$.
    **for** every complex $C_i \in STAR_{new}$ **do**
      **if** (statSignificance($C_i, ES, y$)) and qual($C_i$) > qual($C_{best}$)) **then**
        $C_{best} \leftarrow C_i$
    **let** $STAR$ be the best $n$ complexes from $STAR_{new}$; $n$ is a user-defined size of $STAR$ (usually $n = 5$)
    **let** $STAR^{AB}_{new}$ be such a subset of $STAR_{new}$ where complexes in $STAR^{AB}_{new}$ contain only conditions from $ARG\_REASONS$
    **let** $STAR^{AB}$ be the best $n$ complexes from $STAR^{AB}_{new}$.
    **let** $STAR$ be $STAR$ merged with $STAR^{AB}$.
  **return** rule: "IF $C_{best}$ THEN $y$".