Integration von unvollkommenem regelbasierten Hintergrundwissen in Ähnlichkeitsmaße (Enhancing Similarity Measures with Imperfect Rule-based Background Knowledge)

Dissertation zur Erlangung des Titels
Doktor der Kognitionswissenschaft (PhD in Cognitive Science),
eingereicht am
Fachbereich Humanwissenschaften
der Universität Osnabrück
von

Timo Steffens, M.A.

Betreuer: Prof. Dr. Volker Sperschneider
Prof. Dr. Ute Schmid
Externer Gutachter: Prof. Dr. Ralph Bergmann

Diese Arbeit wurde dem Fachbereich Humanwissenschaften der Universität Osnabrück als Dissertation zur Erlangung des Grades eines Doktors der Kognitionswissenschaften vorgelegt.

Tag der Disputation: 5. Mai 2006
Erstgutachter: Prof. Dr. Volker Sperschneider
Zweitgutachter: Prof. Dr. Ute Schmid
Externer Gutachter: Prof. Dr. Ralph Bergmann

Acknowledgements

A dissertation is never written in isolation, especially not if it is embedded into a program such as the Doctorate Program ”Cognitive Architectures: The Integration of Rules and Patterns” at the Institute of Cognitive Science, Osnabrück. Thus, during my scientific work for this thesis, I benefitted from many people that I would like to thank for their help and support.

First of all, I would like to thank my advisors Prof. Dr. Volker Sperschneider and Prof. Dr. Ute Schmid for embarking into the new lands of Case-Based Reasoning with me. Additionally, many thanks to Prof. Dr. Ralph Bergmann for co-refereeing this thesis and for providing important advice.

The Doctorate Program served as anchor and guideline throughout the last three years. I am indebted to Prof. Dr. Peter Bosch for running the Program and making sure that the other members and I always had all the resources we needed at our disposal. Furthermore, I would like to thank him and the other members of the Doctorate Program for many enlightening discussions about the many facets of cognitive science. More thanks go to Dr. Carla Umbach, the coordinator of the programme, who was always there for guidance and advice in organizational and professional matters.

I would also like to thank Prof. Dr. Claus Rollinger and Dr. habil. Helmar Gust for teaching me my very first steps in Artificial Intelligence and for igniting my interest in Artificial Intelligence research.

I am indebted to Collin Rogowski who wrote parts of the source code that I used for the RoboCup experiments and who provided views from other perspectives.

Special thanks go to Randi Charlotte for providing much needed diversion and for always setting me back on track. Last but not least, many thanks to my parents for their patronage and support.

Abstract

Classification is a general framework that can be applied to various tasks such as object recognition, prediction, diagnosis or learning. There exist at least two different approaches for classification, namely rule-based and similarity-based classification. The two approaches have different strengths and weaknesses. The former requires a domain theory in order to make inferences from the test instance to its class. The latter does not have this requirement and approximates the class of a test instance via its similarity to a set of known instances.

In this thesis the above two approaches are integrated in the realm of Case-Based Reasoning (CBR). CBR treats new cases according to their similarity to stored cases. Similarity is calculated by a similarity measure, which is the crucial factor for classification accuracy. In this work, rule-based domain knowledge is incorporated into the similarity measures of CBR in order to increase classification accuracy. Several novel integration methods are introduced, implemented and evaluated. Since knowledge about real world domains is typically imperfect, the approach does not assume that the domain theories are accurate or complete. Rather, a systematic analysis of different knowledge types is presented that shows the effect of imperfect knowledge on classification accuracy. The analysis is conducted partly empirically in artificial and in real world domains, and partly formally.

Contents

Chapter 1

Introduction

Many cognitive and computational tasks can be seen as classification problems. Object recognition, concept learning, decision processes, diagnosis, and predictions can be treated in a classification framework, to name only a few. Classification can be achieved by rule-based (Anderson, 1983; Newell & Simon, 1990) or instance-based (Kolodner, 1991; Nosofsky, 1984) approaches, which are traditionally assumed to exclude each other. Rule-based approaches process inference rules that express generalized relations between properties. In contrast, instance-based approaches defer generalization until classification time and store instances without further analysis. Generalization is achieved by inferring properties from similar instances. Recently the question arose of how these two approaches can be integrated (Domingos, 1995; Nosofsky, Palmeri, & McKinley, 1994). One motivation for this integration is that there is good psychological evidence that humans apply both rules and instances in categorization and learning (Erickson & Kruschke, 1998). Another, more technical motivation is that both approaches may complement each other in overcoming approach-specific weaknesses (Aamodt, 1994; Hahn & Chater, 1998; Porter, Bareiss, & Holte, 1990; Cain, Pazzani, & Silverstein, 1991).

This thesis builds upon both psychological and technical motivations. More specifically, it is situated in artificial intelligence with a psychological perspective. One goal is to use findings from experiments about human similarity assessment in order to apply them in the technical domain of case-based reasoning (CBR). Since humans are good and flexible at categorization, it is intuitive to test whether cognitive phenomena associated with this task will also be helpful in AI implementations. Another goal is to analyze similarity-based classification from a computational perspective and motivate knowledge-rich similarity measures technically. In other words, in this thesis the influence of domain knowledge on similarity is examined, formalized, and implemented. As will be outlined later, the specific technical aims in this thesis consist of increasing the classification accuracy of CBR systems, and introducing flexibility with respect to different classification goals into these systems. Of course these benefits do not come for free, so an additional aspect is to keep the additional computational overhead low.

Since similarity is an important construct in many domains beyond CBR, the resulting methods and heuristics are formalized in a way that is general enough for use in other fields that employ similarity, e. g. clustering and object recognition.

1.1 Back to the Roots of Case-Based Reasoning

In this thesis, we go back to the roots of case-based reasoning (CBR). In its early stages, CBR was inspired by psychological models about human problem solving: New problems are solved by using experience about similar known problems (Kolodner, 1991). As the state of the art progressed, CBR became practically one of the standard approaches in computer science, with much of the research focussing on technical details. Nowadays, psychological studies provide much more insights about human problem solving and how humans assess the similarity of problem instances (Ahn, Kim, Lassaline, & Dennis, 2000; Choplin, Cheng, & Holyoak, 2001; Medin, Goldstone, & Gentner, 1993). These findings are at odds with typical recent CBR approaches. Our work aims at integrating new psychological models into the similarity measures of CBR in order to enhance their flexibility and accuracy.

CBR works by storing previous cases in a case-base. The cases are represented using a fixed vocabulary of attributes. If a query is posed, the system retrieves the case that is most similar to the query. In some applications, the retrieved case has to be adapted in order to be useful for the query. In classification, such adaption is not necessary, the class of the retrieved case is predicted to be the class of the query.

Creating appropriate similarity measures is considered one of the major challenges in CBR, since the accuracy of a CBR system depends heavily on the similarity measure. Often, the measure is designed by talking to domain experts. This knowledge acquisition is deemed the bottleneck of the CBR designing phase (Craw, 2003). Although it is widely held that domain knowledge is crucial for high performance, a systematic analysis of the different types of knowledge that can be incorporated into a CBR system has not yet been conducted. Even worse, the knowledge acquired by interviewing domain experts is typically imperfect, because many domains are not yet well understood (Porter et al., 1990), and only theoretically uncertain knowledge is available (Aamodt, 1994). We analyze the effect of imperfect knowledge and show empirically and theoretically that even partial, inconsistent and inaccurate knowledge can improve similarity-based classification.

It should be noted that we clearly do not propose a boot-strapping method. We use domain knowledge, but do not focus on acquiring this knowledge by processing the cases. In other words, we are not concerned with learning knowledge bottom-up from the data (refer to (Stahl, 2004) for a recent approach of such a learning mechanism), but we will discuss further below how our approach might facilitate boot-strapping. We assume that there is additional knowledge available, either from outside of the system or from another process. This makes sense both technically and psychologically. In CBR it can be assumed that domain experts possess knowledge about underlying principles that cannot be inferred from a sparse set of cases. In psychology, it can be assumed that humans do not have to infer all knowledge from their own experiences, but acquire socially and culturally shared knowledge.

While we go back to the psychological roots of CBR, we also treat CBR as it was conceived of in the machine learning community originally. In its pure form, CBR is an instance of lazy learning (Aha, 1997). Lazy learning stores the cases without further analysis and defers generalization until classification time. In order to fully exploit the flexibility of lazy learning, a CBR system should not commit itself to a narrow learning bias (Aha, 1997). However, we argue that using a fixed similarity measure (as is the main paradigm in CBR nowadays) is a very rigid learning bias. To soften this bias, we investigate how to dynamically adapt similarity measures to varying classification goals, in order to exploit the strengths of lazy learning. This way, the similarity measure is not fixed anymore, but a function of the classification goal (cf. the Patdex/2-System which adapted weights to the classification goal (Janetzko, Wess, & Melis, 1992)). The system keeps all cases without further analysis in the case-base. The similarity measure is adapted if a new classification goal is used, or if the domain knowledge is updated.

For applications where a wide generalization bias is not necessary (or even counter-productive), there exist CBR approaches that incorporate the advantages of eager learning by generalizing cases already in the case-base (e. g. (Bergmann & Vollrath, 1999)).

1.2 The Knowledge Container Approach in CBR

We follow the knowledge container approach in CBR proposed by Richter (Wess & Globig, 1994; Richter, 1995). It acknowledges that CBR is a knowledge-based approach and states that the existing domain knowledge has to be spread into four knowledge containers: the case representation (vocabulary), the case-base, the similarity measure and adaptation knowledge. More details about these containers will be discussed in the next chapter. In this thesis we focus on how to integrate domain knowledge into the similarity measure container.

According to the knowledge container approach all containers have to be filled for a CBR system to run. Interestingly, almost all knowledge can be moved into an arbitrary container while the other containers are filled only to a minimum. For example, Richter (Richter, 1992) showed theoretically that one can move a maximal amount of knowledge into the similarity measure so that only one case per class is necessary in the case-base. Vice versa, (theoretically) if all possible cases are collected in the case-base, the similarity measure can be reduced to the identity operator abandoning all domain knowledge for similarity (Richter, 1992). Hence, a similarity measure can be seen as a form of knowledge. Bergmann even subsumes similarity assessment in CBR systems under reuse-related knowledge (Bergmann, 2002).

Yet, Richter's analysis does not provide concrete means about how to incorporate knowledge into similarity measures, let alone methods for different types of knowledge or imperfect knowledge.

We examined how rule-based domain knowledge can be exploited for similarity measures. This knowledge can range from single rules to complete domain theories. Since the cases in the case-base can also be regarded as knowledge, we use the term "domain knowledge" to cover knowledge that is not specific to single instances (e. g. *red(car1))*, but refers to relations between attributes (e. g.: *if fast(X) then expensive(X))*.

The direction taken is to analyze the weighting and similarity techniques and then to examine which types of knowledge are useful. We do not follow the reverse direction, which would be to examine which types of knowledge exist and whether and how they can be used in similarity measures. The types of knowledge are too vast to be considered as a whole. Thus, we grouped knowledge types by the way they can be incorporated into similarity measures, and did not group them by epistemological principles.

1.3 Contributions of this Thesis

This thesis provides contributions for both technical and cognitive science questions.

The technical contribution is for similarity-based classification techniques. Traditionally, CBR is viewed as a knowledge-poor method, because it can be applied in domains where no domain knowledge exists. However, during the last 15 years research on integrating knowledge into CBR and similarity measures has been an active area of research. This is due to the fact that in many domains at least partial, inaccurate or inconsistent knowledge exists (Porter et al., 1990; Aamodt, 1990; Bergmann, Pews, & Wilke, 1994; Cain et al., 1991; Surma, 1994). The systematic investigation in this thesis on how knowledge can be exploited will allow developers to adapt or maximize the amount of used knowledge. We developed a systematic hierarchy of knowledge types, introduced incorporation methods for each type and investigated the impact of each type on classification accuracy.

As we mentioned before, we are not considering a boot-strapping method. This means, we do not generate rules or domain theories by processing the case-base. Instead, our approach takes the existing imperfect domain theory for granted and exploits it as much as possible. The systematic analysis of which types of knowledge are useful will lead to incentives to learn such knowledge from the cases or extract it statistically. Then the system would not be dependent of additional knowledge apart from the case-base. Previous work focussed on isolated knowledge chunks which were learned from cases and then used for planning (Armano, Cherchi, & Vargiu, 2004), classification (Gabel & Stahl, 2004; Stahl, 2004), or Bayesian learning (Lamma, Riguzzi, & Storari, 2004). This thesis offers a systematic analysis of various types of knowledge and provides insights into which information should be learned from the instances if the knowledge is not explicitly given.

The contribution to cognitive science is to provide a formal model of how knowledge can be employed in similarity assessment at all. To understand a cognitive system, it is useful to synthesize it, test it, find correspondences to the human paragon, and then examine discrepancies. We analyzed which information is contained in concept hierarchies, rules, isolated definitions, and equations. Although we cannot make claims yet how exactly humans use such knowledge, we can at least show how knowledge can be used in principle. We identified several vague notions of knowledge in the psychological and cognitive science literature that were associated with similarity assessment and clarified them. That is, we distinguish between types of knowledge in a formal and precise way. Furthermore, we grouped these refined types of knowledge in a hierarchy and showed important commonalities and differences. As a side effect this identification of knowledge types might lead to new strategies how to interview domain experts during knowledge acquisition.

With these clarifications of knowledge and the methods to incorporate knowledge into similarity measures it is now possible to derive clear predictions for psychological similarity assessment from our model. This way, we provide a falsifiable model so that it is possible to analyze which parts also hold for human cognition.

To sum up, our main contribution is threefold: First we systematically analyzed which types of knowledge are useful in similarity measures. Second we examined how each type of knowledge can be incorporated into a similarity measure and how it can be exploited. Third, we analyzed which requirements the knowledge must fulfill in terms of completeness, accuracy and consistency.

1.4 Evaluating the Approach

We evaluated the benefit of knowledge-enhanced similarity measures partly formally and partly by simulations in artificial domains and benchmark domains. Since these domains are very different, we believe that our approach is general enough to enhance similarity-based classification in many domains. Furthermore, we implemented the approach also for a specific application, namely similarity-based opponent modelling in multi-agent systems. This application is well-suited for our evaluation purposes, because it allows for various classification goals and is complex enough to experiment with different types of knowledge. The domain is also a particular challenge, since it does not assume that cases are easy to acquire as is assumed in most CBR applications. Instead, it has to be able to make good predictions based on

very few observations.
In brief, opponent modelling tries to predict the behavior of an agent. In the framework of similarity-based classification, the classification goal is the opponent's action and the assumption is that the opponent behaves similar in similar situations.
Opponent modelling is well-suited to show why both instances and general domain knowledge are useful. Domain knowledge (we also used problem solving knowledge) is of a general nature and is thus not detailed enough to serve as a basis for predicting specific opponent actions. In contrast, observations are very detailed, but not general enough to predict actions in other situations. Our approach combines the knowledge contained in observations, i. e. in the case-base, with domain knowledge.

1.5 Related Work

This thesis draws upon several different areas of research. First of all, psychological research on similarity is scattered across different communities, such as analogical reasoning, perception, decision-making, concept formation, and similarity per se. Also in the technical areas, similarity is researched within different communities, such as clustering, object recognition, alignment, information retrieval, and CBR. Thus, it is infeasible to give an exhaustive treatment of related work at this place. Rather, we will discuss related work in the next chapter and give additional references throughout the thesis. For now, we only give a short overview here about closely related work on incorporating knowledge into CBR.
There are also many methods for classification, such as neural networks and regression techniques. However, in our experiments we do not compare our similarity-based approach with those other machine learning methods, because our main focus is the question how domain knowledge can be incorporated into similarity measures, with a cognitive science perspective.
The idea to incorporate rule-based knowledge into CBR is not new. In fact even among the very first CBR systems, CABARET was a system that retrieved legal cases using similarity-based and rule-based methods (Rissland & Skalak, 1989). The purpose of the system is different from our aims: The cases in the case-base were used to sharpen and instantiate vague notions in the background knowledge. For example, a law about work accidents may use the vague notion of "in furtherance of employment" in order to define which

accidents have to be payed for by the employer. Previous legal judgements form cases and are used to instantiate such vague notions. The knowledge was not used to modify the similarity measure.

CABARET is an example for vertical integration (Aamodt, 1994) of rule-based and similarity-based classification. That is, both approaches are intertwined and do not run independent of each other. In contrast, horizontal integration means that two systems run in parallel and individually generate two outputs. A mediator module has to decide which of the two outputs will be the final result.

Our approach is an instance of vertical integration. Prominent further examples of vertical integration are the PROTOS system (Porter et al., 1990) which matches syntactically different features if they are semantically equivalent, and explanation-based CBR (e. g. (Aamodt, 1994; Bergmann et al., 1994; Cain et al., 1991)). In explanation-based CBR, strong domain theories are used to filter attributes that are not used in explanation-chains for the classification.

Our thesis extends previous work substantially, by treating several kinds of knowledge and analyzing the effects of imperfectness, such as partialness, inaccuracy and inconsistency. Furthermore, previous work that incorporated knowledge required structured case-representations. We will show that domain knowledge can be used even for less structured data, namely attribute-value representations.

1.6 Structure of the Thesis

This thesis is organized as follows:

In chapter 2 we give an overview over research on similarity in cognitive science, particularly in philosophy, psychology and computer science, in order to motivate our research. Furthermore, CBR is explained in more detail.

In chapter 3 we introduce the formal notation used in this thesis. Similarity, domain knowledge, and cases are defined.

Based on those definitions, in chapter 4 we define a hierarchy of knowledge types and show how each type can be incorporated into similarity measures.

In chapter 5, we analyze the effect of the knowledge types on classification accuracy. Furthermore, we examine how inaccurate, partial and inconsistent knowledge influences the classification.

In chapter 6 we show how domain knowledge can be combined with weight learning and describe experiments in two benchmark domains. Furthermore, we introduce a method to use domain knowledge in a weight learning method itself.

Chapter 7 extends our methods to similarity-based opponent modelling in multi-agent systems and reports experimental results in the domain of simulated soccer .

Finally, in chapter 8 we conclude and outline future work.

Chapter 2

Similarity in Cognitive Science

> This sense of sameness is the very keel and backbone of our thinking.
>
> *William James, American psychologist (1890)*

2.1 Introduction

Similarity is a prominent notion in cognitive science and its subdisciplines. Already the British Empiricists regarded similarity as the main means of cognition, and psychology researched similarity from the beginning of this discipline throughout to the present day. However, this prolonged work has also resulted in diluting the notion of similarity so that some researchers even see it as an empty shell without meaning and explanatory power (Goodman, 1972). Furthermore, due to the cyclic progress in science, many insights about similarity have been ignored for a long time. In this chapter we define our own starting point by reviewing previous work and identify challenges and open questions from a cognitive science perspective. We discuss similarity in philosophy and psychology, and computer science techniques that are related to similarity-based classification.

2.2 Similarity in Philosophy

The most simple notion of similarity was used by the British Empiricists, such as Locke and Hume. According to them, the only way to increase one's insight into the world is by experience. Experience comes as so-called "ideas", that is, objects of perception. Composite ideas (e. g. *chair*) are compounded from simple ideas (e.g. *brown* or *wooden*). Ideas are then associated using only three principles of connection: Contiguity in place and time, cause and effect, and resemblance (Hume, 1777/1975) (which in fact is our notion of similarity).

This means that one can predict similar effects given similar causes. Resemblance was based on sharing components or so-called sensory qualities. Qualities would be called dimensions or attributes nowadays. Resemblance was clearly distinguished from matching. Whereas the former requires sharing of similar components, the latter requires sharing of strictly identical components. It is obvious that resemblance is a recursive definition of similarity, since similarity of an "idea" is based on the similarity of its components. The ability to perceive the similarity of primitive ideas (e. g. *brown* and *red*) was assumed to be inherent in human cognition. In the view of the Empiricists, similarity is a bottom-up phenomenon, directly determined by the perception apparatus. There is no top-down influence of cognition and knowledge.

Similarity was thought of as a binary relation. Either two "ideas" were similar or not. Similarity was not graded as is usually the case nowadays. Furthermore, similarity was based on superficial attributes, that is, on properties ("qualities") that can be sensed directly. Context or even differences in perception were regarded as irrelevant.

When philosophy gave birth to psychology, Bain (1855) extended the ideas of associationism with a thorough investigation of the physiology of sensation and movement. However, he does not define similarity more clearly than the British Empiricists, but bases it on the undefined primitive *resemblance*. The ability to perceive such resemblance seems to be directly given to humans (Jurisica, 1994).

In Carnap's Aufbau (1928) all binary reflexive and symmetric relations were similarity relations. Relations that were additionally transitive are equivalence relations. He set out to define all scientific concepts using similarity relations on so-called raw experiences. This is again reminiscent of the British associationism.

It was Fullerton (1890) who took Bain's idea that similarity is upper-bounded by identity to grade similarity between dissimilarity and identity. However, the semantics of graded similarity remained unclear. Goodman (Goodman, 1951) dealt with this problem by using a four-place predicate to state that a pair of objects is at least as similar to each other than another pair.

Goodman entertained a sceptic view on similarity, though. He rejects the idea of objective similarity independent of language. Two objects can be viewed as similar only if they are referred to by the same language constructs. This means for example, that two objects that are believed to belong to the same concept in language are viewed as similar after this concept has been identified. Goodman sharpened his criticism in a famous article (Goodman, 1972) where he states that similarity is a "quack" and denies it any explanatory power. Since similarity is no absolute notion, the statement that two objects are similar yields no information at all unless one lists the features that they share. This, however, is also useless since any two objects share infinitely many features. An illustrative example from (Hahn & Chater, 1998) is that a lawnmower and a plum have in common that they weigh less than 100 kilo, and less than 101 kilo and so on. Thus, all objects are equally similar to each other. From a philosophical point of view Goodman's criticism was a rigid point against similarity as an explanatory construct for cognition. Only later, the criticism was dissolved by work in psychology.

2.3 Similarity in Psychology

Early psychological work on similarity was heavily influenced by the philosophical tradition of associationism. This is not surprising, because the first psychologists were often philosophers who studied the human mind. The work of Bain mentioned above can also be seen as psychological research. Also the foundational work of William James sees a foundational role of similarity in cognition, because along with contiguity it forms the foundations of associations (James, 1890).

Although the idea of the British Empiricists had an influence on early psychological work, the new discipline provided new research methods, namely empirical experiments. This resulted in a purely descriptive notion of similarity as grouping principle for Gestalts (Wertheimer, 1923). The Law of Similarity states that objects are perceptually grouped together if they look similar.

The main area of interest at that time was psychophysics, that is, perception.

Similarity appeared as basis of generalization in primary stimulation gradients (Pavlov, 1927). In experiments on conditioning it turned out that the probability of a reaction to a new stimulus is a function of its similarity to the old stimulus. Similarity was defined as the distance on sensory dimensions such as tone frequency or loudness.

Although there has been work on similarity for a long time, the first modern study on psychology (Jurisica, 1994) is considered to be Wallach's article on psychological similarity in the late 50s (Wallach, 1958). He identifies several weaknesses in the assumptions of the British Empiricists and pits "psychological" similarity against their "potential" similarity. The latter can be seen as a form of objective physical similarity, whereas the former is perceived similarity. That is, Wallach assumes that humans can select or ignore features. He also reports experiments that showed effects of context and attention. Furthermore, most important for this thesis, he argues that also extrinsic features might play a role in similarity assessment. Such features are not perceivable directly, but have to be attributed from the outside to the object, such as its use or its construction. This already hints to the hypothesis that knowledge (e. g. about what an object is used for) has to be activated for similarity assessment. In this sense, similarity also has top-down aspects.

After Wallach's article, research on similarity widened its focus and different various aspects were examined. To account for this, in the remainder of this chapter we will not treat the literature chronologically anymore, but thematically.

Common to the different psychological approaches is that they treat similarity between representations and not between the real objects.

2.3.1 Object Representation

Similarity can be researched only if some assumptions on the representation of the objects or cases are made, because similarity always works on representations (Medin & Ortony, 1989).

Today, there are three main approaches of measuring psychological similarity. They differ in the way objects are represented. The geometrical model has been introduced by Shepard (1957). The set-theoretic contrast model was proposed in Tversky's classic paper (Tversky, 1977). Relational or structured similarity is advocated by Gentner (1989).

Shepard's model treats similarity of objects as their distance in a psychological space spanned by the objects' attributes.

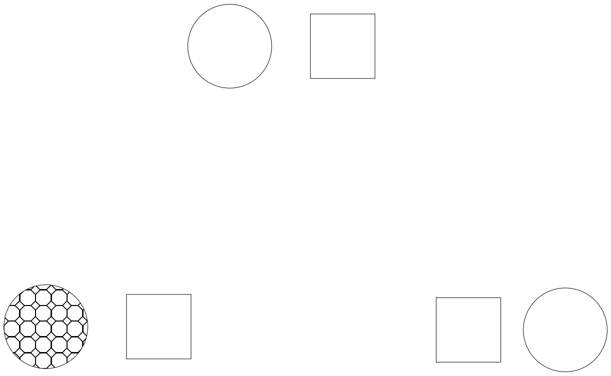

Figure 2.1: Three cases of arranged geometrical objects showing the necessity of relations. Only a relation can express that in the rightmost arrangements the circle is to the left of the square. Adapted from Medin, Goldstone, Gentner, 1993.

In Tversky's contrast model, similarity is a function of shared and distinctive features of the comparison partners. Features are assumed to be discrete, and are compared via an identity operator. This simple feature approach does not allow to cope with context or different classification goals, which resulted in experimental observations that suggested that reflexivity and symmetry of similarity are invalid.

Both Shepard's and Tversky's approaches assume very simple representations for objects, either as points in an n-dimensional space or as sets of discrete features. However, most cognitive science theories for representation of natural objects - such as faces, visual scenes, or sentences - assume structured representations (Hahn, Chater, & Richardson, 2003). Objects usually consist of subcomponents or are related to other objects in taxonomical or mereological relations. Therefore, much research has gone into structural similarity (Gentner, 1989; Markman & Gentner, 2005; Hahn & Chater, 1998; Markman, 2001). In the structural approach, perceivable features are represented as unary predicates. The crucial idea is to additionally use relations with higher arity. This way, it cannot only be stated which features an object has, but also how features are interrelated. For example, by using a "left of" relation, it can be stated in a case of geometrical arrangements that an object A is to the left of an object B (see figure 2.1). Such a statement is impossible in simple feature representations.

Hahn and Chater (Hahn et al., 2003) propose an even more general approach to similarity that subsumes the afore-mentioned models as special cases. They suggest that similarity between two objects is determined by the complexity required to transform the representation of one objects into the representation of the other object. Based on the mathematical notion of Kolmogorov complexity, their approach can handle arbitrary kinds of representation. However, to make falsifiable predictions, their model requires selection of a finite set of primitive transformations. Up to now it is not clear what transformations can be seen as given and whether additional transformations can be learnt.

2.3.2 Respects

Goodman noted that similarity is a useless notion, since the statement that two things are similar bears no information unless one states in which respect they are similar (Goodman, 1972). That is, one has to list the features in which they are identical. As already mentioned in section 2.2 this argument remained an important criticism for a long time. However, Medin, Goldstone and Gentner (1993) suggest that the respects of similarity are fixed systematically. The respects arise from the task, context, background knowledge, and from the objects themselves. Thus, respects constrain the (according to Goodman theoretically infinitely many) features that enter the comparison process and thus also determine similarity.

In Medin et al.'s experiments there was high agreement between subjects for similarity ratings. This is a contradiction to Goodman's statement that similarity is arbitrary and carries no information. It is hypothesized that the across-subject agreement in similarity ratings is due to the fact that the features that are used for comparison are constrained systematically (Medin et al., 1993). In the extreme, children are very rigid about similarity, since they do not analyze stimuli in their components but compare them as wholes (L. B. Smith, 1989).

But also adults show systematic selection of features. Context activates features or makes them less salient. For example, a snake and a racoon are judged to be more similar if the context is *pet*, than if no context is provided. This reminds of Goodman's idea that objects that both belong (or do not belong) to a concept are judged more similar than if they belong to different concepts (cf. 2.2).

Research in analogy suggests another phenomenon that determines which features are used for similarity assessment. Similarity between problems is assumed to determine whether analogical transfer from a known problem to a new problem is successful. Apparently, isolated features are less likely to enter the comparison process than features that are interrelated and belong to a system of connected relations (Gentner, 1989). This phenomenon is called systematicity and is the core of Gentner's structure mapping theory of analogy and alignment (Gentner, 1983). There is good evidence that the success of analogical transfer depends on the type and size of the structural overlap of such relational systems (Schmid, Wirth, & Polkehn, 2003). The theory also predicts that n-ary relations are more likely to enter the comparison process than unary predicates (object properties).

Further constraints on the selected features come from process principles of comparison (Medin et al., 1993). For instance, it has been reported that the properties of the object that is encountered first are more likely to enter the comparison process than the properties of the second object (Tversky, 1977). Context, systematicity, and process principles complement each other to constrain the respects of similarity. It is however uncertain, whether these constraints are strong enough to completely fix the respects of similarity. It has been proposed to see the flexibility of similarity not as a weakness, but as a strength, though. People can dynamically adapt their similarity assessment to various tasks and contexts without the need to relearn features or feature relevance (Lamberts & Chong, 1998).

2.3.3 Background Knowledge

In this section we review psychological work on the question whether and how background knowledge influences similarity assessment. We look at how knowledge provides additional attributes, and how knowledge helps to determine feature relevance.

Abstract attributes:

In some domains generalization cannot work on superficial attributes alone (Wallach, 1958). Not all properties of an object that are necessary for classification are directly perceivable. For example, in chess it is not due to their shape that the bishop is more similar to the rook than to the pawn, but by their ability to move or their number in the starting configuration (Groot, 1978). Apparently, such properties have to be inferred or retrieved from memory.

For such inferred properties, Medin and Ortony coined the term of "deep similarity", as opposed to "surface similarity" which is based on readily accessible components (Medin & Ortony, 1989). Surface (or superficial) attributes can be represented by simple unary predicates (such as $blue(X)$) (Medin et al., 1993), whereas for deep similarity more complex relations with higher arity are needed (such as $causes(X, Y)$). Obviously, abstract relations require knowledge (Brown, 1989; Gentner, 1989). For example, children compare a sponge and a cloud by mere appearance as "both are round and fluffy", while adults as "both hold water and give it back later" (Gentner, 1988). This difference is due to better representation and structuring of knowledge (Carey, 1984).

This is also in line with the finding that the comparison processes of novices and experts differ in that the former rely on superficial attributes, whereas the latter use more abstract attributes (Chi, Feltovich, & Glaser, 1981). For example, in order to determine the similarity of physics problems, physics novices use superficial features like the objects mentioned in the problem description (e. g. "river steamer" or "wooden box"). In contrast, physics experts determine problem similarity based on underlying principles like "conservation of energy". Such abstract attributes are only present if some knowledge has already been acquired (Chi et al., 1981; Medin et al., 1993), because they have to be inferred from the superficial attributes that are contained in the problem description.

Furthermore, a dissociation exists between categorization and surface similarity (Ahn & Dennis, 2001). That is, people may judge an object A more similar to B than to C, but still categorize A identical to C. However, this dissociation vanishes if subjects are encouraged to use deep similarity. This suggests that categorization is based on deep rather than surface similarity.

Feature relevance:

Considering that in theory all objects share infinitely many features, there have to be some criteria to decide which features are important and which are irrelevant. Furthermore, some features will be more diagnostic than others. For instance, when deciding whether a particular animal is a bird or a mammal, the feature *has feathers* is more diagnostic than *can fly* (Lamberts & Chong, 1998), because both bats (being mammals) and eagles (being birds) can fly.

Experiments about similarity judgements suggest that features that cause other features are deemed more important than features that are effects of or only correlated to other features (Choplin et al., 2001). In an experiment subjects were told that keys with a certain notch were able to open a particular safe. The opening of the safe is an effect of the notch in the key. Additionally, keys had the feature whether they were able to open the safe. In the similarity ratings it turned out that the feature *notch* (cause) influenced similarity between keys more than the feature *open safe* (effect). The higher relevance for cause features provides support for the hypothesis that domain knowledge about the relations between attributes influences similarity assessment.

This finding is also supported by another study in which subjects learned a causal chain (Ahn et al., 2000): "Because Roobans have sticky feet, they can climb trees. Because they can climb trees, they eat fruits." The experiments showed that the feature that is the first cause in the chain (*sticky feet*) is the most important feature in the similarity rating, followed by the feature that is the intermediate cause (*climbs trees*). Finally, the feature that is only an effect (*eats fruits*) had the smallest impact on similarity.

Furthermore, the experiments suggest that causal background knowledge can be used to infer missing or unobservable features (Ahn et al., 2000).

However, feature relevance is not static, but is assessed dynamically based on the task and the context (Lamberts & Chong, 1998). In an experiment subjects were asked to categorize faces into families based on their similarity to other members of the families. They assigned different relevance to features depending on whether they were told that the faces in comparison were related as cousins or as brothers. It has been proposed that these differences are due to background knowledge that is activated by the concepts of cousin and brother (Lamberts, 1994). What is most relevant for this thesis is that feature relevance can be rapidly changed by giving additional information. If subjects are told that a particular feature is relevant for the target category, they adjust their similarity assessment immediately by increasing the influence of that feature (Lamberts & Chong, 1998). This suggests that feature relevance is not (or at least not only) slowly learned, but can be set by isolated chunks of knowledge. Furthermore, similarity is not determined by bottom-up processes, but can also be influenced by top-down processes.

Another phenomenon related to feature relevance is salience. Even if a feature might be objectively relevant, it is not certain that it will be selected, unless it is salient. Salience of features for conceptual similarity is regarded to depend on the activation of knowledge (Medin et al., 1993). It is infeasible to assume that all knowledge related to an object will be activated for comparison. Instead, only properties that are strongly associated with a stimulus will be activated upon presentation of the stimulus and will influence similarity. This introduces a great deal of flexibility and ambiguity into similarity judgements, because background knowledge varies across subjects and knowledge activation depends on the context. For example, if the similarity of England and the United States is to be judged, in the context of sport the concepts soccer, basketball and football might be activated (Medin et al., 1993). In the context of geography, knowledge about continents might be activated. Hence, it has to be known in which respects the similarity of objects should be tested in order to determine which knowledge chunks are activated and are used as attributes for similarity.

2.3.4 Categorization

Similarity is believed to play an important role in categorization. In fact, the technical approach of similarity-based classification is based on psychological models of categorization (Kolodner, 1993; Ross, 1989).

Categories and concepts are essential for cognition, since they allow to generalize over experiences and to infer predictions (Bruner, Goodnow, & Austin, 1956). For example, if an object is categorized as *tiger* based on its color and shape, further attributes such as *dangerous*, *carnivorous*, and so forth can be predicted (Lamberts & Chong, 1998).

For a lot of categories no necessary or sufficient conditions can be given (Rosch & Mervis, 1975). Therefore it was proposed that objects are categorized based on their similarity to prototypes (Rosch & Mervis, 1975) or on their similarity to other exemplars (Nosofsky, 1990). Even for categories that can be defined in terms of necessary and sufficient conditions, it might be the case that humans categorize them similarity-based in daily life. For example, whales were defined to belong to the category of mammals, whereas people might classify them as fish because of their shape and living situation.

Although theory-based (rule-based) approaches to categorization have been proposed due to dissociations between categorization and similarity (see (Rips, 1989) or (Ahn & Dennis, 2001) for a discussion), we will review only similarity-based approaches here.

Barsalou (Barsalou, 1989) reports that category representations are unstable. The representation of the color *red* varies depending on whether it is used together with *wine*, *hair*, or *apple*. He explains this by the hypothesis that category representations consist of an context-independent core, a context-dependent part, and a part of recently activated knowledge. Features in the core are always activated if the category is used. Features in the context-dependent part are only activated in certain contexts. Finally, some features are only activated if they were recently used. Thus, the similarity of categories will vary across contexts and the recent experiences of the subject. However, the question remains open which features belong into which part of the category representation (DeJong, 1989).

As noted earlier, a dissociation between categorization and *surface* similarity was found by Ahn (Ahn & Dennis, 2001). However, she reported a correlation between categorization and *deep* similarity. Apparently, for categorization other attributes are used than for mere surface comparisons. This is also supported by findings in analogy (Gentner, 1983).

However, category complexity seems to determine whether classification is done rule- or similarity-based. Various experiments suggest that similarity-based classification is only used if the target rule, which separates positive from negative instances, becomes complex[1]. Otherwise, rule-based processes seem to be activated (see (E. E. Smith, Patalano, & Jonides, 1998) for an overview over such studies). It was also shown that similarity-based classification is useful for handling exceptions (Erickson & Kruschke, 1998).

Further psychological research confirmed theoretical hypotheses that similarity cannot be used for extrapolation tasks: Subjects that are instructed to press a button with a duration proportional to the size of a stimulus, are able to extrapolate their response to extremely long stimuli that have not been encountered before (Shanks, 1995). A similarity-based model of classification cannot account for this phenomenon.

The above findings suggest that the interaction of rule- and similarity-based processes takes place on a very high level. That is, both processes work in parallel (Erickson & Kruschke, 1998; E. E. Smith et al., 1998). However, there is also evidence for interaction on earlier levels. One area of psychological research puts forward the argument that similarity-based classification cannot be done without rule-based knowledge at all: as mentioned in section 2.3.3 the assessment of similarity is strongly influenced by rule-based knowledge about feature relations: In experiments conducted by Ahn and Kim (Ahn et al., 2000), attributes that causally influenced other attributes were more important in similarity judgements than effect-attributes.

[1] Complexity of a rule is operationalized as the number of critical attributes that need to be checked.

2.4 Computer science in general

Similarity assessment is a common method in computer science. Its use is most prominent in nearest-neighbour classification (Friedman, 1994), case-based reasoning (Kolodner, 1993) and information retrieval (Yang, 1999). Typically, similarity is used in applications for which no perfect domain theory or general knowledge exists (Porter et al., 1990). The assumption in all similarity-based classification approaches is that the target function is smooth. That is, similar objects are classified similarly. This assumption is also used in an attempt to give a semantic to similarity values. The similarity between two objects is in some frameworks seen as probability that the two objects belong to the same class (Richter, 2003). In other application areas apart from classification, the similarity measure is an a-priori heuristics about the actual similarity. The latter can only be measured after the solution has been generated, so that it is an a-posteriori criterion (cf. (Stahl, 2004)).

Several general-purpose similarity measures have been developed. None of them is optimal in the sense that it performs better than the others in all domains (Griffiths & Bridge, 1997; Cover & Hart, 1967). Choosing a similarity measure depends on the object representation such as attribute-value representations, graph representations, predicate logic representations, and object-oriented representations. For an overview see (Bergmann, 2002). We will discuss the first two in more detail, since they are commonly used.

Often, distance functions are used instead of similarity measures. However, it is widely accepted that a similarity measure s can be derived from a distance measure d via $s = 1 - d$.

2.4.1 Attribute-value representations

The simplest form of representation is the attribute-value representation. An object is a set of attributes, and each attribute has a type.

For numeric attributes, the most common measure is the Minkowski norm (here normalized into the interval [0,1], cf. (Bergmann, 2002)):

$$d_{Minkowski,p}(x,y) = \left(\frac{1}{n} * \sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$$

where x and y are the objects to be compared, n is the number of dimensions, x_i and y_i are the values for dimension i of object x and y, respectively. The

Minkowski norm is a distance metric, that is, it is symmetric, reflexive and fulfills the triangle inequality. The parameter p can be chosen so that the metric resembles the City Block Metric ($p = 1$), Euclidean Distance ($p = 2$), or the Maximum norm ($p \rightarrow \infty$).
For binary attributes, the best known measure is the Hamming Distance. It measures the distance of two objects based on the number of attributes in which they differ (Langley & Iba, 1993):

$$d_{Hamming}(x, y) = \frac{1}{n} * |\{i | x_i \neq y_i\}|$$

Attributes can also be weighted. We will introduce weighting in chapter 3.
Note that the representation methods are not mutually exclusive. The attribute-value representation can be transformed into an analytical representation, i. e. vectors (which will be discussed below), and into a logical representation as unary predicates (Richter, 1992).

2.4.2 Graph representations

In many domains, objects cannot be represented as consisting of several independent dimensions. For example, a machine consists of many connected parts and subparts. Graphs can be used to capture the object's hierarchical structure. For example, parts can form the nodes, and the *has-part* relationship is specified by edges between the nodes.
Assessing the similarity of graphs is computationally expensive (Bergmann, 2002). In the graph matching approaches, determining the largest common sub-graph is NP-complete (Mehlhorn, 1984).
Where determining the largest common sub-graph has the advantage that similarity can be defined in a continuous way, checking graph isomorphism or sub-graph isomorphism yields a binary similarity measure. The former has a factorial worst-case complexity, the latter is NP-complete (Bergmann, 2002).
Another way to determine graph similarity is graph-editing (Bunke & Messmer, 1994). Reminiscent to the approach of string editing, similarity is determined by the cheapest set of editing operations that need to be performed in order to transform one graph into the other. The editing operations *insert node*, *insert edge*, *delete node*, *delete edge*, *change node label*, *change edge label* have an assigned cost, and the cost of the cheapest sequence of operations

determines similarity. Also calculating this measure is NP-complete (Bunke & Messmer, 1994).

Note that the psychological model of Gentner's structural similarity (see 2.3.1) is a graph representation.

2.4.3 Token vectors

In Information Retrieval, objects represent text-documents (e. g. (Klinkenberg, 1998)). They are represented as vectors, where each component specifies the frequency of a token (i. e. word) in the document. For example, the vector $[4, 0, 1, \ldots]$ means that the first token (e. g. "house") appears 4 times in the document, the second token does not appear at all, the third token appears once, and so on.

Similarity between documents can be defined as the cosine of the angle between document vectors in the vector-space (Manning & Schuetze, 1999). Typically, the components are weighted by the relation between the token's frequency in the document and in the whole document collection.

2.5 Case-Based Reasoning

Case-Based Reasoning (CBR) was inspired by human problem-solving and categorization (Kolodner, 1991). In a way, CBR is the computer science incarnation of psychology's exemplar-theory (Nosofsky, 1990). New problems are solved by exploiting experience with known problems. That is, a case-base stores previously seen cases which will be compared to the new problem. The most similar known case is retrieved from the case-base. The retrieved solution is then adapted for the new problem.

This paradigm can be used for problem-solving (Wilke & Bergmann, 1998), diagnosis (Baumeister, Atzmueller, & Puppe, 2002), prediction (Nunez et al., 2002), and planning (Bergmann et al., 1994; Rodriguez, 2001). In this thesis, we are particularly concerned with classification, which is a general enough framework to subsume or at least overlap with diagnosis and prediction.

2.5.1 Strengths of CBR

How does CBR differ from other machine learning methods such as decision trees, neural networks, and Bayesian learning?

In its pure form, CBR is an instance of lazy learning (Aha, 1997), which means that training instances are stored without further analysis. Generalization and bias are deferred until classification time, making CBR flexible enough to use the same training instances (cases) for different purposes. For example, in the Patdex/2 system, attribute weights were dependent on the class (Janetzko et al., 1992). Further, the generalization and bias can be adapted for the given query. However, in most systems this flexibility is traded-off against performance. Similarity is typically determined by a fixed, optimized measure (Wettschereck, Aha, & Mohri, 1997). So generalization is not deferred until classification time, but already fixed at design or implementation time.

Nowadays, CBR systems are typically a combination of eager and lazy learning. For example, cases can be combined to generalized cases (Bergmann & Vollrath, 1999), which means that analysis and generalization take place before classification-time.

To further exploit the advantages of lazy learning, this thesis analyzes how similarity measures can be adapted to classification goals. We show how knowledge that is dependent on the classification goal can be incorporated into similarity measures.

Compared to neural networks and reinforcement learning, CBR performs well with sparse data and usually needs much less training instances. Furthermore, CBR is robust against noise, since a noisy case has only a local effect.

Disadvantages of CBR compared to pure eager learning methods are the high storage requirement and the fact that the generalization takes place at the query time. Fortunately, improvement of the similarity measure allows one to reduce the case-base size (Wess & Globig, 1994). Another problem is that the generalization is performed at classification-time, which is often time-critical. Thus, the flexibility with respect to classification-goals is traded off with higher computational effort at classification-time.

2.5.2 The CBR cycle

A widely used framework for describing the CBR cycle has been proposed by Aamodt and Plaza (1994). It divides the main process into four subprocesses (the so-called ”four REs”): the retrieval, reuse, revise and retain processes (see figure 2.2). After presenting a new problem (called the query) to the system, the retrieval process finds the most similar case from a case-base. Usually this involves some form of indexing so that not the whole

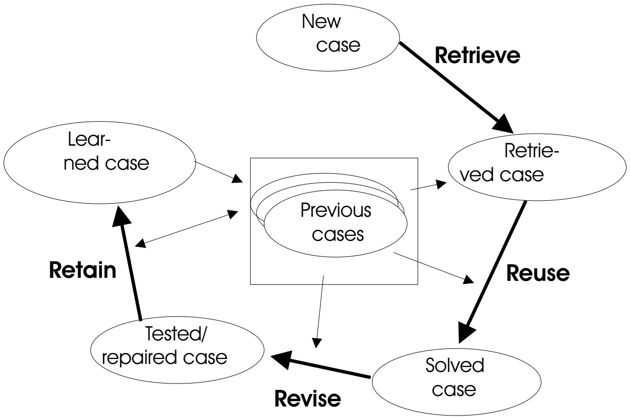

Figure 2.2: The CBR cycle as proposed by Aamodt and Plaza (1994).

case-base has to be processed. The reuse process exploits the information (often called the solution) contained in the retrieved case to solve the query. In the classification framework reuse is particularly easy as it returns the class of the retrieved case as the class for the query. In other applications (e. g. planning or problem-solving) the revise process adapts the old solution so that it fits to the new problem. Adapting retrieved solutions is often not trivial and is an active area of research (e. g. (Boerner, 1994; Wilke & Bergmann, 1996, 1998)). Finally, the retain process determines whether the query and its solution should be stored into the case-base. This decision takes into account how costly the adaptation of the solution was and how densely the case-base is populated in the neighborhood of the query. In the retain phase, additional learning mechanisms can be used (see (Stahl, 2004) for an overview).

In this thesis we focus exclusively on the retrieval process, because it makes use of similarity measures to determine the most similar case for a query. Reuse and revise methods are mostly trivial for classification. Thus, the retrieval process is the main concern for classification with CBR. Indexing techniques can be used to speed up retrieval but are also not in the scope of this thesis.

2.5.3 Knowledge containers

A complementary view to Aamodt and Plaza's framework is the knowledge container approach by Richter (Richter, 1995). In this approach, the similarity measure, the case-base, the vocabulary of the case-representation, and the adaptation methods are seen as knowledge. In order to implement a CBR system, each of the four knowledge containers has to be filled at least minimally. The knowledge can be shifted from one container into any other (Wess & Globig, 1994). For example, if all knowledge is moved into the case-base container, so that the case-base (theoretically) contains all possible cases, then the similarity measure can be simplified to the identity operator (Richter, 1992).

Moving knowledge from the similarity measure to the case-base has no effect on the classification for a given classification goal (Richter & Althoff, 1999). For example, a rule like $x < 30$ can be integrated into the similarity measure by making instances more similar that both satisfy or both do not satisfy the rule. But it can also be incorporated into the case-base by removing all cases that do not satisfy it. We argue that removing cases contradicts lazy learning, since it performs eager generalization. Instead, we propose to move as much knowledge as possible into the similarity measure in order to increase flexibility. This way, the system keeps all cases in the case-base and can adapt the similarity measure if a new classification goal is used, or if the domain knowledge is updated. This is particularly useful if there is no overabundance of data (as for example in our evaluation domain of opponent modelling).

When designing and implementing a CBR system, domain experts are consulted in order to acquire the knowledge necessary to fill the knowledge containers. For example, if the CBR system will be used for diagnosing machine faults, the domain experts are interviewed about the attributes that are contained in a fault report (e. g. the temperature of and the pressure in the machine) and their value range.

These attributes are part of the vocabulary container, just as their value range or even taxonomies. The vocabulary defines which information of the domain is important. In this sense, the vocabulary is the basis or language to describe the knowledge in the other containers. Defining the vocabulary is a crucial task when building a CBR system.

The experts might also specify how similarity between fault reports should be calculated. Although it is not common to interview experts for concrete attribute weights, they might be asked which of the attributes are particularly important. In many systems a lot of knowledge is needed for the adaptation of the retrieved solutions (e. g. (Bergmann & Wilke, 1998)). Since the CBR system designers are rarely experts in the domain, adaptation rules have to be provided by domain experts.

The knowledge container approach is particularly interesting for this thesis, because it shows that similarity measures are a form of knowledge and can be extended with additional knowledge. Furthermore, domain experts are in the loop anyway. Thus, it is reasonable to assume that some knowledge exists even in domains where no perfect domain theories are available. It is promising to analyze how partial, inaccurate or even inconsistent knowledge can be exploited in similarity measures.

2.5.4 Incorporating knowledge into CBR

The idea of interfacing CBR with background knowledge is not new. Much effort has been done to use domain knowledge for adapting retrieved solutions (e. g. (Bergmann & Wilke, 1998; Bergmann, Wilke, Vollrath, & Wess, 1996; Wilke & Bergmann, 1996)). We will not discuss those but focus on approaches that deal with similarity measures.

The motivation for enhancing similarity assessment with domain theories is the following: While cases are usually represented in a case language which specifies superficial and intrinsic attributes, generalization needs more than such superficial attributes in most domains (Porter et al., 1990). Abstract attributes are needed to generalize in terms of functions, roles, and relations. Such abstract attributes are defined in domain theories.

Explanation-based CBR (EBCBR) uses inference rules to create so-called explanations describing why a solution is appropriate for a given case (Bergmann et al., 1994; Cain et al., 1991). If an attribute was not used in the explanation, it is regarded as irrelevant and ignored in future similarity assessment. Thus, the motivation is to filter irrelevant attributes, whereas in our approach the aim is to find additional attributes that help to approximate the utility of solutions via similarity.

Another branch of EBCBR uses explanations to "explain away" differences that are either irrelevant or only of a syntactical rather than semantical nature (Aamodt, 1994). Similarly, while not regarded as EBCBR, the famous PROTOS system (Porter et al., 1990) uses domain-knowledge for matching syntactically different features, too. The main difference to our approach is

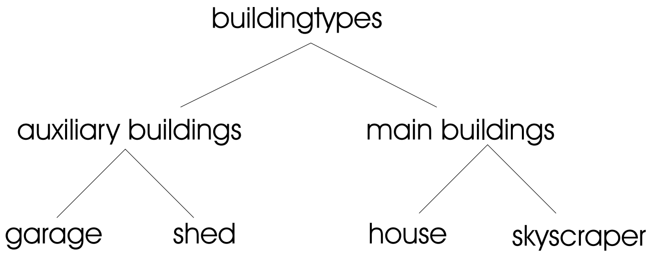

Figure 2.3: A simple concept hierarchy.

that those methods process only attributes that are already explicitly represented in the cases.

Adding abstract attributes to the similarity measure is similar to completion rules, which are often used in diagnosis tasks (Wilke & Bergmann, 1996). As the name suggests, completion rules calculate abstract attributes on the basis of other attributes. They are a general means to integrate rule-based domain knowledge.

Several types of knowledge are used in (Rodriguez, 2001). The similarity measure is influenced by a concept hierarchy and a network of causal relations. The closer two concepts (i. e. values of a nominal attribute) are in the concept hierarchy, the higher their similarity. For example, assume a case has a discrete nominal attribute *buildingtype* (see figure 2.3). If the values *garage* and *shed* are under the same father concept *auxiliary building*, which does not subsume *skyscraper*, then *garage* is more similar to *shed* than to *skyscraper*. Moreover, if two concepts stand in a causal relationship or have identical effects according to the causal network, their similarity increases.

Formerly, it was proposed that rules can be used for CBR only in two ways, namely for "term reformulation" and "case elaboration" (Branting & Porter, 1991). The former replaces an open-textured term by one or more terms that are less open-textured. Case elaboration on the other hand infers features that are only directly specified in one case, but not in the other, or that subsume mismatching features. In our hierarchy of knowledge types (see chapter 4) these two ways are subsumed under the same type, namely matching knowledge. Since they cope only with matching, they are limited to features that are already in at least one of the cases. This is of no use in homogenous case representations, e. g. in chess board configurations or simulated soccer situations, where all cases are represented with the same fixed set of attributes

(Steffens, 2004a).

Besides the afore-mentioned approaches that deal with individual types of knowledge explicitly, use of domain knowledge in recent CBR systems typcially takes place in an ad-hoc and application-specific way (Aamodt, 2001) (refer to the INRECA-II methodology (Bergmann, Breen, Goeker, Managao, & Wess, 1999) for a first step towards a general framework for defining knowledge-rich similarity measures, though).

Obviously, traditional use of domain theories in CBR misses the possibility to infer attributes that are explicit in none of the cases. Constructive induction (CI) was concerned with changing instance representations by creating informative features or transforming existing predicates (e. g. (Fu & Buchanan, 1985; Matheus, 1991; Rendell, 1989; Gunsch & Rendell, 1991)). While the idea is closely related to our thesis, the focus is different. CI was mainly concerned with rule-based induction (see (Aha, 1991b) for an exception), while in our work we examine how similarity-based classification can be supported. Furthermore, we take a top-down perspective, while CI is a bottom-up approach.

All of these CBR approaches that incorporate knowledge used structured representations. In this thesis we show that domain knowledge can also be exploited if the cases are represented as attribute-value lists.

2.5.5 Knowledge Acquisition

Research on knowledge acquisition in general is mainly concerned with specifying and sharing ontologies. The notion of ”ontology” is not clearly defined, though. Interpretations range from a system that contains the vocabulary of a logical system to a full semantical account of a domain (refer to (Guarino & Giaretta, 1995) for an overview).

An ontology is different from a domain theory, since the former is an abstraction of the latter. That is, an ontology can refer to several different domain theories. The ontology provides a viewpoint on a domain theory which can be parameterized (Schreiber, Wielinga, & Jansweijer, 1995). For example, the exact values in the domain theory can be adjusted by its father ontology.

Knowledge acquisition in CBR views a CBR system on the knowledge level (Aamodt, 2001), an abstract level which describes the goal and purpose of the system, and the tasks that have to be performed. The following three types of knowledge are proposed: task knowledge, method knowledge, and domain knowledge. The task knowledge defines a hierarchy of tasks and subtasks (Motta, Fensel, Gaspari, & Benjamins, 1999). For instance, the task "diagnose car" can be decomposed into "observe symptoms", "decide tests", "perform tests", and "identify likely faults" (Aamodt, 2001). Method knowledge then describes how these tasks can be executed. The methods use domain knowledge in order to solve the tasks. Such domain knowledge has been proposed to contain facts, heuristics, causal relationships, and specific cases.

In contrast to this thesis, knowledge acquisition for CBR systems does not acquire domain knowledge to be used in similarity measures. Instead, domain knowledge is used mainly for adaptation, the vocabulary, and to capture cases (Leake & Wilson, 1999).

In its beginning, CBR was considered a knowledge-light approach (Wilke, Vollrath, Althoff, & Bergmann, 1997). Nowadays there is the branch of knowledge-intensive CBR (Diaz-Agudo & Gonzalez-Calero, 2001), which acknowledges the importance of domain knowledge. Here knowledge acquisition plays a role. Yet, knowledge acquisition in CBR is different from knowledge acquisition in general. A major research area of the former is to acquire cases (Strube, Enzinger, Janetzko, & Knauff, 1995) and to reuse components of implemented CBR systems. The work in this thesis may be regarded as a first step to bridge knowledge acquisition in general with knowledge acquisition in CBR.

There exist advanced tools for knowledge management for CBR designers, such as CBR-Works (Bergmann, Breen, et al., 1999) and CBROnto (Diaz-Agudo & Gonzalez-Calero, 2001) which integrates ontologies and CBR. It allows the designer to incorporate knowledge into the case representation by suggesting commonly used descriptors like *has-part*, *has-precondition*, *has-effect* depending on the application such as planning, diagnosis, or prediction. The retrieval process can be guided by selecting one of the standard similarity measures for attributes depending on their type. CBROnto provides some domain-independent operators for solution adaptation (such as *delete part* or *insert part*) and tries to learn domain-specific adaptations from the adaptations that the domain experts execute during the training phase.

In this thesis, we represent knowledge more in the spirit of general knowledge acquisition (as opposed to knowledge acquisition in CBR). Domain knowledge is specified by domain theories. Our definition of domain theory is based on logical Horn clauses and is given in chapter 3.

2.6 Conclusions and Motivations

In this chapter we have given an overview over research on similarity in subdisciplines of cognitive science. To conclude, we summarize the aspects that are most relevant for this thesis.

A major insight is that similarity is not an absolute notion. Instead, the respects in that objects are similar need to be specified. In humans, these respects are systematically fixed by the context, task, and classification goal. This systematic fixing also means that respects can be flexibly adapted to the classification needs. We take this as motivation to make similarity measures dynamically adaptable to the classification goal and show where this flexibility is beneficial.

There is overwhelming evidence that humans use knowledge to guide similarity assessment. Similarity is both a bottom-up and a top-down process, influenced by knowledge. This is in stark contrast to the use of ”knowledge-light” similarity in computer science. Humans use deep attributes that are activated by background knowledge. This way, they are able to compare and categorize objects in more meaningful ways than just superficial attributes. A large part of this thesis deals with how to incorporate such deep attributes as additional attributes into similarity measures.

But humans do not only use knowledge for constructing features, they also use it for determining the relevance of features. For example, causal relationships boost cause attributes over effect attributes. This works even if humans have available only isolated chunks of knowledge. Thus, we will take into account that domain knowledge can be useful for similarity-based classification even if it is imperfect.

We have also discussed various object representations in psychological and computational models. For our approach we adopt the attribute-value representation since it is general enough for many CBR applications, and particularly well-suited for our evaluation application of opponent modelling in multi-agent systems as we have shown in (Steffens, 2004a, 2005d) (also cf. (Ahmadi, Keighobadi-Lamjiri, Nevisi, Habibi, & Badie, 2003; Wendler, 2004)).

The main approaches to psychological similarity (geometric, set-theoretic and structural) neglect the issue of finding relevant properties and weights (Hahn & Chater, 1998). This thesis proposes that this can (only) be done by using background knowledge in form of rules and provides a first computational model.

Finally, Hahn states that researching the influence of different types of knowledge on similarity remains an important open question (Hahn & Chater, 1998). We also tackle this question and provide a hierarchy of knowledge types and show how they can be used for technical similarity measures.

Chapter 3

Definitions: Cases, Similarity Measures, Domain Theories

In this chapter we define the basic concepts that are used in our approach, such as attributes, cases, domain knowledge and similarity measures.

3.1 Attributes and Cases

Informally, ”an attribute, in brief, is any discriminable feature of an event that is susceptible of some discriminable variation from event to event.” (Bruner et al., 1956) (p. 26). Formally, we define:

Definition 3.1.1. *A set* $A = \{a_1, a_2, \ldots, a_n\}$ *is an* attribute*. The* a_i *are called* attribute values*.*

Definition 3.1.2. *An attribute* A *is* nominal *iff* $|A|$ *is finite. Among the attributes with* $|A| = \infty$*, we call an attribute* numerical*, iff it is an interval* $[a, b]$ *with* $a, b \in \mathbb{R}$*.*

Definition 3.1.3. *The* range *of a numerical attribute* A *is* $range(A) = (\arg\max_x x \in A) - (\arg\min_y y \in A)$*.*

We assume that for all attributes $range(A) \neq \infty$.

Definition 3.1.4. *An* attribute set $\mathbb{A}$ *is a set of attributes* $\{A_1, A_2, \ldots, A_n\}$*.*

Definition 3.1.5. *The* universe $U(\mathbb{A})$ *over the attribute set* $\mathbb{A} = \{A_1, A_2, \ldots, A_n\}$ *is* $U(\mathbb{A}) = A_1 \times A_2 \times \ldots \times A_n$ *in an arbitrary but fixed order.*

Definition 3.1.6. *A predicate $f : U^m \to A$ is a* classification *iff $m = 1$, f is defined on the whole U, and A is a nominal attribute. If furthermore $|A| = 2$, we call the class* binary*.*

That is, a classification yields a (complete and disjunct) partition of the universe of instances. We do not treat graded class membership.
Note that the mathematical notion of a relation arises if $m > 1$ and $|A| = 2$. In the remainder we will sometimes refer to binary classes as *concepts*.

Definition 3.1.7. *A* case *or* instance *$c \in U(\mathbb{A})$ with $|\mathbb{A}| = n$ is an n-tuple of attribute values $(v_1, v_2, \ldots, v_n)$, where $v_i \in A_i$ and $A_i \in \mathbb{A}$. We call $A_i(c) = v_i$ the case's* value *for A_i.*

In other words, we assume homogenous cases, that is, all cases have the same attributes. Furthermore, we assume that there are no missing values (except the target class). This is for example different from CBR work in diagnosis (Baumeister et al., 2002) or medicine (Porter et al., 1990).
When designing a CBR system, the representation (i. e. the choice of attributes) should be classification distinguishable. That is, two cases whose classes are different need to be distinguishable. Note that this assumption is hard to guarantee in practical realistic implementations where no perfect domain models exist. Thus, there is a trade-off between including enough attributes to be classification distinguishable and avoiding irrelevant attributes. Note that $A(c)$ for a nominal attribute A is also a class, and for a numerical attribute $A(c)$ is a function.

Definition 3.1.8. *Given a universe $U(\mathbb{A})$ with $\mathbb{A} = \{A_1, A_2, \ldots, A_n\}$, an $n - 1$-tuple $q = (a_1, a_2, \ldots, a_{t-1}, a_{t+1}, \ldots, a_n)$ with $a_i \in A_i$ and $0 \leq t \leq n$ is a* query*. $A_t \in \mathbb{A}$ is called the* target attribute*.*

A query is a case for which the target attribute is not known. Regarding the target attribute, in traditional CBR cases are often treated as pairs (d, l), where d is an element of the so-called description space and l an element of the lesson- or solution-space (Bergmann, 2002). Target attributes are elements of the lesson-space. In our approach, there is no such dichotomy between description- and lesson-space, because it would contradict the principle of lazy learning that cases are stored without further analysis. The target attribute is determined at classification time by providing a query. In other words, one attribute of a query is not known and has to be predicted from the other attributes.

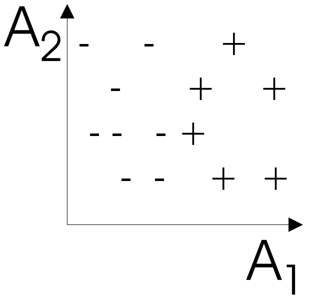

Figure 3.1: A universe spanned by the attributes A_1 and A_2, and cases from a case-base. + denotes cases that are members of the target class, and - denotes non-members.

In this thesis, the target attribute A_t is always nominal, since we treat classification. But similarity-based techniques are also applicable for regression, which assumes numerical target functions (Atkeson, Moore, & Schaal, 1997).
A *case-base* I is a set of cases, that is $I \subset U(\mathbb{A})$.
A case-base for a two-dimensional universe is depicted in figure 3.1.

3.2 Similarity

The proposed methods for incorporating knowledge do not rely on similarity measures that are based on a particular form of distance function (such as Manhattan distance), but are general enough to work with most similarity measures, no matter what distance function they use. However, for consistency throughout the chapters, our experiments and analyses were done with a particular similarity measure. This also means that we do not cope with knowledge about which measure to choose. This question, like the question of the distribution of values, has been worked on in the statistical literature (see (Atkeson et al., 1997) for references).

Definition 3.2.1. *A case $c' \in U(\mathbb{A}')$ with $\mathbb{A}' = \{A_1, A_2, \ldots, A_{t-1}, A_{t+1}, \ldots, A_m\}$ is a* projection *of case $c \in U(\mathbb{A})$ with $\mathbb{A}' = \{A_1, A_2, \ldots, A_m\}$, iff $|\mathbb{A}'| = |\mathbb{A}| - 1$, $\mathbb{A}' \subset \mathbb{A}$, and $A_i(c') = A_i(c)$ for $1 \leq i \leq m, i \neq t$.*

In the remainder, we will often not distinguish between cases and their projections, as the projection can be easily performed and the context should make it clear which one is meant.

Definition 3.2.2. *A* local similarity function $d_i(c_n, c_m)$ *with* $0 \leq i \leq |\mathbb{A}'|$ *is defined as* $U(\mathbb{A}') \times U(\mathbb{A}') \to [0, 1]$.
In particular,
if $|A_i| \neq \infty$,

$$d_i(c_n, c_m) = \begin{cases} 1 & : \ iff \ A_i(c_n) = A_i(c_m) \\ 0 & : \ else \end{cases}$$

(which is the dual to the Hamming distance).
If $|A_i| = \infty$,

$$d_i(c_n, c_m) = 1 - \left(\frac{|A_i(c_n) - A_i(c_m)|}{range(A_i)} \right)^2$$

Note that one of the cases will usually be a query and the other one a projected case from the case-base.
A local similarity function calculates the similarity of two cases on one attribute. We handle nominal attributes (i. e., attributes with a finite domain) only via identity. In some other works, nominal values can be ordered (e. g. (Surma, 1994)). Then the similarity between different pairs of nominal attribute values can be different, so that for example *pink* can be more similar to *red* than to *green* (assuming that there is an attribute $A = \{pink, red, green\}$). In contrast, according to our definition, comparing *pink* to *red* yields a similarity of 0, just as comparing *pink* to *green*.

Definition 3.2.3. *A* similarity measure $s(c_n, c_m)$ *is defined as* $U(\mathbb{A}') \times U(\mathbb{A}') \to [0, 1]$,

$$s(c_n, c_m) = \sqrt{\frac{1}{m} \cdot \sum_{i=1}^{m} (w_i \cdot d_i(c_n, c_m))}$$

where $c_n, c_m \in U(\mathbb{A}')$, $m = |\mathbb{A}'|$ *and* $0 \leq w_i \leq 1, \sum w_i = 1$. *The* w_i *are called* weights*.*

The similarity measure is a weighted sum of the local similarity functions, excluding the target attribute. For numerical attributes, the similarity measure is equivalent to the Weighted Euclidean Metric (cf. (Bergmann, 2002)) due to our definition of local similarity for numerical attributes.

Definition 3.2.4. *A similarity measure is called* reflexive*, iff* $\forall c : s(c, c) = 1$.

Definition 3.2.5. *A similarity measure is called* symmetric*, iff* $\forall c_1, c_2$: $s(c_1, c_2) = s(c_2, c_1)$*.*

Although the general form of our similarity measure is symmetric, in later chapters we will also consider measures that are not symmetric due to contextual attributes that depend on the query.

Definition 3.2.6. *A similarity measure satisfies the* triangle inequality*, iff* $\forall c_1, c_2, c_3 : s(c_1, c_2) + s(c_2, c_3) \leq 1 + sim(c_1, c_3)$*.*

If we describe similarity in form of distance (i. e., $d(c_1, c_2) = 1 - s(c_1, c_2)$), the triangle inequality has the more commonly used form: $d(c_1, c_2) + d(c_2, c_3) \geq d(c_1, c_3)$.

The perfect similarity measure with respect to target attribute A_t would be $sim(c_1, c_2) = 1$ iff $A_t(c_1) = A_t(c_2)$.

3.3 Classification

3.3.1 The classifier

A classifier's purpose is to predict the class of a query c. In other words, a classifier predicts the unknown value $A_t(c) \in A_t$ for a query c. In CBR and k-nearest neighbor it is assumed that all misclassifications have equal cost (Wettschereck et al., 1997).

Remember that we will often not distinguish between cases and their projections, as the projection can be easily performed and the context should make it clear which one is meant.

Definition 3.3.1. $Q(c_1, s, I)$ *is a* candidate set *of query* c_1 *with a similarity measure* s *on a case-base* I*, where* $Q(c_1, s, I) \subset I$*. The elements in* $Q(c_1, s, I)$*, called* candidates*, satisfy the condition* $\forall c_2 : (c_2 \in Q(c_1, s, I) \rightarrow c_2 \neq c_1 \wedge \neg\exists c_3 \in I : s(c_1, c_3) > s(c_1, c_2))$*.*

That is, the candidate set for case c_1 contains those cases from the case-base I that are most similar to c_1.

Definition 3.3.2. $T(c_1, s, I) \rightarrow A_t \cup \{ambigue\}$ *is a* classification *of query* $c_1 \in U(\mathbb{A}')$ *with a similarity measure* s *on a case-base* $I \subset U(\mathbb{A})$ *with respect to the target attribute* A_t *with* $\mathbb{A} = \mathbb{A}' \cup \{A_t\}$*.* $T(c_1, s, I)$ *yields* ambigue*, iff*

$\exists c_2, c_3 \in Q(c_1, s, I) : A_t(c_2) \neq A_t(c_3)$, *and yields* $a_i \in A_t$, *iff* $\forall c_2 \in Q(c_1, s, I) : A_t(c_2) = a_i$.
A classification is called correct *with respect to class* f, *iff* $T(c_1, s, I) = f(c_1)$, *and* wrong *otherwise.*

A classification is ambigue if the classification candidates have different values for the target attribute. If all classification candidates (often there is only one) have the same value for the target attribute, that value is returned to be the classification for the query. If this value is the same that the to be approximated class returns, the classification is correct.

3.3.2 Generalization bias

The definition of similarity for nominal and numerical attributes has of course consequences for classification.
For nominal attributes the classification bias is that instances on the same axis-parallel hyper-planes are with a high probability in the same class. The more hyper-planes are shared, the higher the probability.
For numerical attributes the classification bias is that the closer (as defined by the similarity measure) instances are in the instance-space, the more probable is that they belong to the same class.

3.4 Ontological/epistemological considerations about domain theories

Just as we assume that the cases in our universe $U(\mathbb{A})$ are assumed to reflect objects in the real world, and that the attribute set $\mathbb{A}$ can approximate the objects' real-world properties, we assume that we can approximate the structure and regularities of the target class f. If f is defined randomly and without structure, it can hardly be represented in any approach. Thus, our assumption is that there is some structure in terms of causal relations and correlations between properties. We call this knowledge about the structure of the target function domain knowledge. Domain knowledge does not refer to specific instances or cases, but rather to attributes and their relations.
The real world's structure can in general only approximately be described by a formal language (see figure 3.2). However, to specify domain knowledge we need another specification language apart from the case representation

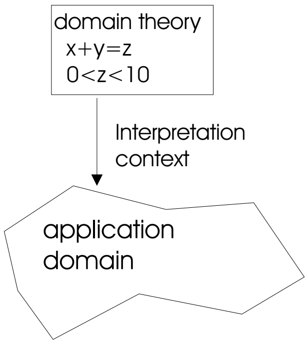

Figure 3.2: A domain theory is connected to the application domain via an interpretation context. Only parts of the domain can be modelled by the theory. Adapted from Schreiber, Wielinga and Jansweijer (1995).

language. As noted in (Schreiber et al., 1995), such a language can be seen as a certain view on the world. The properties of the real world are interpreted by terms of the formal language.
For the domain knowledge language, one has to commit oneself to some ontological assumptions. For example, in the language that we use in this thesis, the difference between causality and inference is lost. The next section defines the formal language for domain knowledge.

3.5 Definition of the domain theory language

Definition 3.5.1. *A is an* intermediate *in the universe $U(\mathbb{A})$, if A is an attribute and $A \notin \mathbb{A}$.*

Intermediate concepts are predicates that are used in the domain theory and are not used to represent the cases in the case-base. If we write *attribute* in the following, it may either be a case attribute or an intermediate.

Definition 3.5.2. *$A(c)$ is an* arithmetic term *if A is a numerical attribute and $c \in U(\mathbb{A})$.*
r is an arithmetic term if $r \in \mathbb{R}$.
If α and β are arithmetic terms, then $(\alpha + \beta)$ is also an arithmetic term.
If α and β are arithmetic terms, then $(\alpha - \beta)$ is also an arithmetic term.
If α and β are arithmetic terms, then (α / β) is also an arithmetic term.
If α and β are arithmetic terms, then $(\alpha \cdot \beta)$ is also an arithmetic term.

If α and β are arithmetic terms, then (α^{β}) is also an arithmetic term.

For sake of readability we will avoid brackets if no ambiguity is introduced by doing so.
A formula is an expression that can be assigned a truth-value:

Definition 3.5.3. *If f is a concept, then $f(c)$ is a formula.*
$(t_1\ OP\ t_2)$ is a formula, if t_1 and t_2 are arithmetic terms, and $OP \in \{<,>,<=,>=,=,\neq\}$.
$(A(c) = a)$ is a formula, if A is a nominal attribute, $c \in U(\mathbb{A})$ and $a \in A$.
$(\phi_1 \wedge \phi_2)$ is a formula, if ϕ_1, ϕ_2 are formulas.
$(\phi_1 \vee \phi_2)$ is a formula, if ϕ_1, ϕ_2 are formulas.
$(\neg\phi)$ is a formula, if ϕ is a formula.

If $|A| = 2$, we call A a binary attribute. For brevity, we will often use A as a proposition in rules, assuming without loss of generality that one of the two values is treated as "true" and the other as "false".

Definition 3.5.4. *A* rule *ρ is either a function of the form $\rho : f(c) = \alpha$, where f is a function, $c \in U(\mathbb{A})$, and α is an arithmetic term, or a concept of the form $\rho : f(c) \leftarrow \phi$, where f is a concept, $c \in U(\mathbb{A})$, and ϕ is a formula. $head(\rho) = f$ is also called the* head *of a rule.*

Note that rules are unary but can have n-ary relations in their body.
Let us consider an illustrating example in the domain of mushrooms. Assume that the cases in the case-base represent mushrooms that are members or non-members of the class *chanterelle*. Assume that they are represented by a simple set of attributes {*height, capThickness, color*}. Let *height* be numerical in the interval [0.5,20], *capThickness* numerical in the interval [0.1,2], and *color* be nominal with the domain {*white, red, yellow*}. A case could look like $(10, 1.2, yellow)$. An example for a function rule is $stemHeight(c) = height(c) - capThickness(c)$. A concept rule could look like $Lachnocladiaceae(c) \leftarrow ((height(c)/stemHeight(c)) > 100) \wedge (color(c) = yellow)$.

Definition 3.5.5. *A* theory *Ψ is a set of rules.*
$C(\Psi) \subset \Psi$ denotes the set of concept rules in Ψ and $F(\Psi) \subset \Psi$ denotes the set of function rules in Ψ.

It is possible that the structure in the real world cannot be captured by a theory of that type. However, in order to have tractable theories we committed ourselves to the language specified here.
In principle it is possible that the knowledge is not formulated in several inference rules (in the form of a tree) but in one long rule using discjunctions and conjunctions. In such a case, strictly speaking there are no intermediates. To remedy this problem, there exist feature construction approaches that modify and transform domain theories (Fawcett & Utgoff, 1992). For example, conjunctions might be candidates for intermediates and can be moved into a new rule. This is especially sensible if a conjunction appears more than once in the rule. However, such situations should be rare, since domain experts (or people who would be able to give a long specification of a concept) are known to have well structured domain-knowledge (Medin et al., 1993). So, in short, we assume a domain theory in conjunctive normal form that can be viewed as a tree.

Definition 3.5.6. *The* intermediates *of a theory Ψ are denoted by $Int(\Psi)$.*

Given a rule ρ in a theory Ψ, $head(\rho)$ is an element of $Int(\Psi)$.
In the literature on domain theories, case attributes are referred to as observables (Mooney & Ourston, 1991), intermediates as intermediate concepts, and the target attribute as classification goal. When the domain theory is depicted as a tree (see Fig. 3.3), the observables are located at the bottom, the classification goal at the top, and the intermediates are in between.
The language here has no capabilities to talk about individual objects or cases. Instead, it is attributes and their relations that are specified. This is due to our motivation to incorporate general domain knowledge into similarity measures. "General" means that the knowledge is not specific to individual cases but about attributes.

Definition 3.5.7. *The* direct conditions $con_d(\rho)$ *of a concept rule* $\rho : f(c) \leftarrow \phi$ *are those attributes that appear in* ϕ.
The direct conditions $con_d(\rho)$ *of a function rule* $\rho : f(c) = \alpha$ *are those attributes that appear in* α.
An attribute A_i *is in the* transitive conditions $con_{t,\Psi}(\rho)$ *of a rule* ρ *with respect to a theory* Ψ*, if* $(A_i \in con_d(\rho)) \vee (A_k = head(\rho_2) \wedge A_i \in con_d(\rho_2) \wedge (A_k \in con_d(\rho) \vee A_k \in con_{t,\Psi}(\rho)))$.

Obviously, $con_d(\rho) \subseteq con_{t,\Psi}(\rho)$. In the mushroom example above, the direct conditions of *Lachnocladiaceae* are $\{$*height, stemHeight, color*$\}$ and the

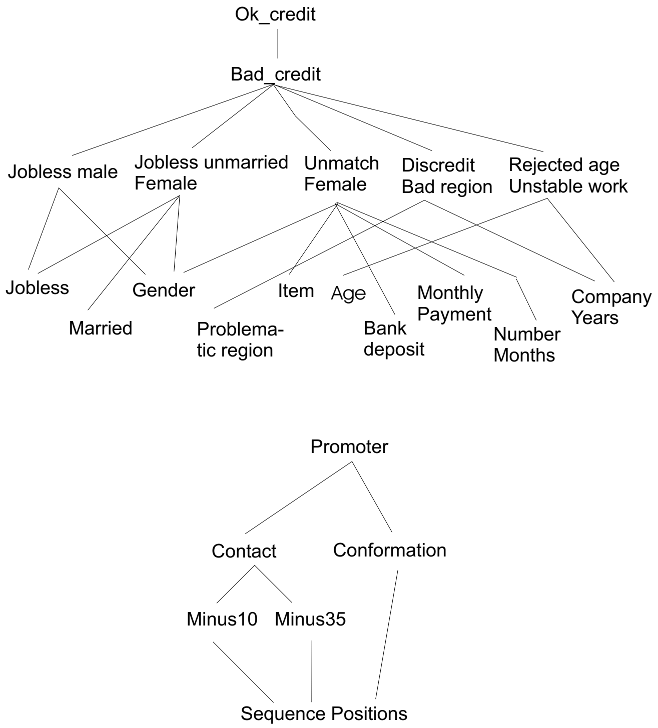

Figure 3.3: Domain theories of two domains. Nodes denote attributes, arcs denote that the more general attribute is defined in terms of the less general attributes.

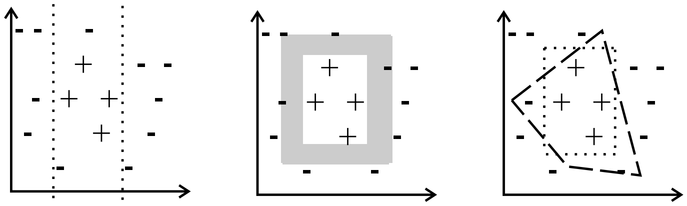

Figure 3.4: Properties of domain theories. The theories describe parts of the target concept, of which there are positive (+) and negative (-) instances. Left: Partial knowledge, only parts of the concept boundaries are known. Middle: Vague knowledge, concept boundaries are believed to be somewhere within the shaded areas. Right: Inconsistent knowledge, different rules make differing predictions.

transitive conditions are {*height, stemHeight, color, capThickness*}.

3.5.1 Informal description of imperfectness

As motivated in section 2.5.5 knowledge about a domain is typically imperfect. There exist at least the following types of imperfectness (see Figure 3.4) which we have already described informally in (Steffens, 2004b):

- Partialness: This is the case if some parts of the domain are not modelled. Examples:
  - Conditions are used but not defined. In figure 3.3 this would be the case for the attribute *jobless male*, if the arcs between this node and *jobless* and/or *gender* were removed.
  - The relation of intermediates or directly represented case attributes (observables) to the classification goal (target attribute) is not known. In figure 3.3 this would be the case for the attribute *contact*, if the arc between this node and *Promoter* was removed.
  - The classification goal (target attribute) does not exist in the rulebase at all. For example, if the node *Promoter* and the corresponding arcs were removed in figure 3.3, there would be no information on how the observables and intermediates were related to the classification goal.

These situations correspond to gaps at the "top" or "bottom" of the domain theory (Mooney & Ourston, 1991).

Partial knowledge does not always mean that the knowledge is incomplete for use. For example, if you know that a certain action is optimal in a situation, it is not necessary to know the detailed effects of another action in that situation. Hence, if we talk of partial knowledge in the following, we mean "partial wrt. to the task".

In machine learning, work on partial domain theories usually deals with the task to complete the theory by means of induction from experimental evidence (e. g. (Bergadano & Giordana, 1998)). However, in this thesis, we do not try to complete the theory, but take it as it is.

- Vagueness and resulting inaccuracy: This is the case if values can only be given within a certain confidence interval. In domains with vague knowledge, classifications do not follow with logical strictness, but can only be tentative. Rules may be about correlations or use confidence factors. Vagueness of knowledge can for example be represented as a mean with a confidence threshold. The correct value is within this threshold with a certain percentage, say 95%. Such a representation cannot be used in a straight-forward way in a similarity measure. Thus, we handle vagueness in the following way: We assume a single value is selected (for example, the mean or the one with the highest probability of being correct). Then this value is used throughout the similarity calculation. This reflects the fact that vague knowledge typically results in the adoption of inaccuracies. In our experiments and formal analysis we investigate the impact of the degree of inaccuracy on the classification accuracy.

- Inconsistency/Alternative theories: In many science domains there exist alternative theories which make different predictions and it is not known which theory is correct. This is typically reflected on the level of rules, where different definitions for the same intermediate exist. Often this is handled as disjunction in a single theory so that it is not obvious whether a class is disjunctive or has alternative theories (e. g. the Promoter Gene domain in the UCI Machine Learning Repository (Blake & Merz, 1998)). CBR is often used to overcome this problem, because the cases provide knowledge which classification is correct for individual cases.

Although we will show later how imperfect domain theories can be exploited for similarity-based classification, there are extreme domain theories that are not suitable for our approach at all. Since our approach processes cases, domain knowledge is only useful if it can be related to the exemplars. Thus, if the domain theory has so many gaps at the bottom that no intermediate or target attribute can be related to the case attributes, the theory is useless. Also, if the attributes in the theory are named differently than the corresponding attributes in the cases, the theory cannot be used at all. Note that we are aware of the possibility to use a bridging language between the case attributes and the theory attributes. But we view such a bridging language as part of the domain knowledge, and thus as part of the domain theory.

While extensive gaps at the bottom of domain theories are disadvantageous, gaps in the top of domain theories can be treated more easily. Since gaps of the latter type mean that the relevance of an attribute to the classification goal is not known, such gaps can be bridged by weight learning methods. We will treat such methods in a later chapter.

In the next section we give formal definitions for types of theory imperfectness.

3.5.2 Definition of imperfectness

Partialness:

Definition 3.5.8. *A rule ρ is* syntactically partial *(we write $partial_{syn,\mathbb{A},\Psi}(\rho)$) with respect to a set of attributes $\mathbb{A}$ and a theory Ψ, iff $\exists A : A \in con_{t,\Psi}(\rho) \wedge A \notin \mathbb{A} \wedge (\neg \exists \rho_2 \in \Psi : head(\rho_2) = A \wedge \neg partial_{syn,\mathbb{A},\Psi}(\rho_2))$.*

This means that a rule is syntactically partial, if there is an attribute in its transitive conditions that is neither an observable, nor defined by a non-partial rule.

Definition 3.5.9. *A theory Ψ is* semantically partial *iff there exists $A_1, A_2 \in Int(\Psi) \cup \mathbb{A}$ so that in the real world there is a dependency between A_1 and A_2, and $A_2 \notin con_{t,\Psi}(A_1)$ and $A_1 \notin con_{t,\Psi}(A_2)$.*

Definition 3.5.10. *A theory Ψ is* partial *if it contains at least one syntactically or semantically partial rule or if $A_t \notin Int(\Psi)$.*

We have two syntactic ways of checking whether a theory is partial. If a rule uses undefined conditions or if the classification goal is not contained in the

theory, it is partial. There is also a semantic way: If the theory does not specify a dependency which exists in reality, then the theory is partial.
This of course raises the question about dependencies that are stated in the theory but do not exist in the real world. We subsume this by the definition of inaccuracy. Superfluous dependencies is the problem of irrelevant attributes which is handled by weight learning (see chapter 6). Superfluous dependencies are easier to fix than missing dependencies. This is intuitively clear, since filtering an irrelevant attribute by setting its weight to 0 is easier than reasoning about an attribute that does not exist in the representation.

Inaccuracy:

Definition 3.5.11. *A rule ρ is incorrect iff* $\exists c \in U(\mathbb{A}) \wedge A = head(\rho) \wedge A(c) \neq \rho(c)$.

Informally, if a rule predicts a wrong value for a case, then it is incorrect.

Definition 3.5.12. *A rule is* inaccurate *if it is incorrect and not partial.*

In other words, an inaccurate rule contains (different from partial rules) at least the correct attributes in its condition, but is incorrect in the constants or operators or contains too many attributes. For example, if the real world is best approximated by the rule $f(c) = A_1(c) \cdot 3 + A_2(c)$, an inaccurate rule would be $f'(c) = A_1(c)/A_2(c)$, because both rules use the attributes A_1, A_2. As discussed above, we do not treat rules that have associated confidence factors or rules that use values with a confidence interval.

Definition 3.5.13. *A theory is* inaccurate *if it contains at least one inaccurate rule.*

Inconsistency:

Definition 3.5.14. *A theory Ψ is* inconsistent*, if* $\exists \rho_1, \rho_2 \in C(\Psi) : head(\rho_1) = head(\rho_2) \wedge \exists c \in U(\mathbb{A}) : \neg(\rho_1(c) \leftrightarrow \rho_2(c))$*, or* $\exists \rho_1, \rho_2 \in F(\Psi) : head(\rho_1) = head(\rho_2) \wedge \exists c \in U(\mathbb{A}) : \rho_1(c) \neq \rho_2(c)$.

In short, a theory is inconsistent if there are two or more rules that predict different values for the same attribute.

Chapter 4

Types of knowledge

> Again within the field of
> judgement itself we find
> varieties, knowledge, opinion,
> prudence, and their opposites;
> of the differences between these
> I must speak elsewhere.
>
> *Aristotle, On the Soul, Part III*

4.1 Introduction

In this chapter we introduce and define several types of knowledge that can be incorporated into similarity measures. Some of these types were used under different names in psychology, artificial intelligence and cognitive science for different reasoning methods. We show how they can be used in similarity-based classification. Furthermore, we arrange them into a hierarchy and show parallels between knowledge types that previously were studied in isolation.

Our grouping of knowledge types is different from previous work. Other typifications of knowledge in CBR are based on the size, strength, or environmental dependency (Althoff & Aamodt, 1996).

The types of knowledge that can be represented are of course constrained by the domain theory language specified in section 3.5. Still, the language proves to be general enough in order to represent a variety of knowledge types.

We do not treat frames, scripts, extended logics or other classical knowledge representation methods, because the main focus of this thesis is to show that even for attribute-value representations domain knowledge can be useful. Classical knowledge representation methods require more structured data. Neither do we group the knowledge types by epistemological principles. Instead, our analysis comes from the direction to examine similarity to find out which sorts of knowledge are useful for similarity-based classification. In other words, we group knowledge types from the perspective how they can be incorporated into similarity measures. This can lead to new strategies about how to interview domain experts during knowledge acquisition and in which form their knowledge should be written down. Furthermore, a systematic analysis of which types of knowledge are useful will provide insights into which information should be extracted from the instances if interviewing domain experts is not feasible.

For all the different types of knowledge, we make no assumptions as to the knowledge's perfectness. Rather, in our approach we assume that we get chunks of knowledge, which might be inaccurate, incomplete, or inconsistent as outlined in section 3.5.1, and try to exploit it as much as possible.

We will first review several knowledge types that have been used in CBR and in psychological work on similarity under various names. Differences and commonalities will be pointed out so that each knowledge type can be defined. Furthermore, a hierarchy of these knowledge types is proposed (see figure 4.1). The semantics of the hierarchy are that all incorporation methods for a knowledge type can also be applied to its subtypes. We will describe the relation between knowledge types in the following sections.

We will show how the types are related to each other and we point out new incorporation methods for each of them. To our knowledge this is the first approach that formally defines knowledge types that can be incorporated into similarity measures (apart from our earlier work in (Steffens, 2005d) and (Steffens, 2005b)).

4.2 Types of knowledge proposed in CBR

In this section we identify the types of knowledge that have been used in CBR. Typically, knowledge has not been regarded in terms of types, but rather each approach used knowledge in its own ad-hoc and domain-specific way. In most cases, the implementations used hybrids of different types of knowledge.

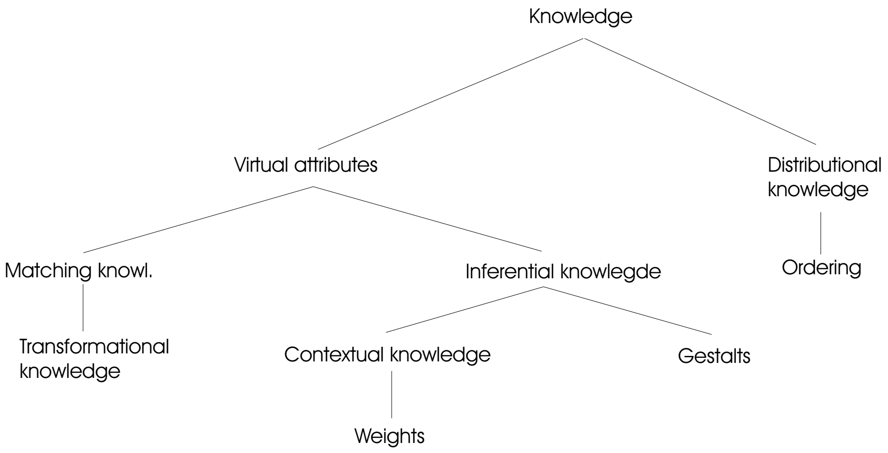

Figure 4.1: Hierarchy of knowledge types.

This section contributes to a systematic analysis which role knowledge plays in CBR and how the various kinds of knowledge can be arranged into a common framework. As a result, knowledge types that were previously seen as isolated will be handled by the same incorporation methods.

Remember that in the knowledge container approach cases are also a form of knowledge. There is also work on how to capture experience in form of cases from domain experts (e. g., (Leake & Wilson, 1999)). However, as noted before we do not investigate case-specific knowledge but focus on general domain knowledge about the attributes and their relations to each other.

4.2.1 Virtual attributes

Virtual attributes (Richter, 2003) are attributes that are not directly represented in the cases but can be inferred from the already existing attributes. This requires inference rules which are part of domain knowledge.

Virtual attributes are useful if the monotonicity-principle is violated. If $s(c_1, c_2) > sim(c_1, c_3)$ is necessary to reflect class membership, then there must at least be one attribute A_i, so that $d_i(c_1, c_2) > d_i(c_1, c_3)$. If such a pair does not exist, the similarity measure must make use of interdependencies between attributes. For example, the similarity may not depend on two attributes A_1, A_2 themselves, but on their difference $A_1 - A_2$. Virtual attributes can express such interdependencies (e. g., $deposit(c) = income(c) - spending(c)$).

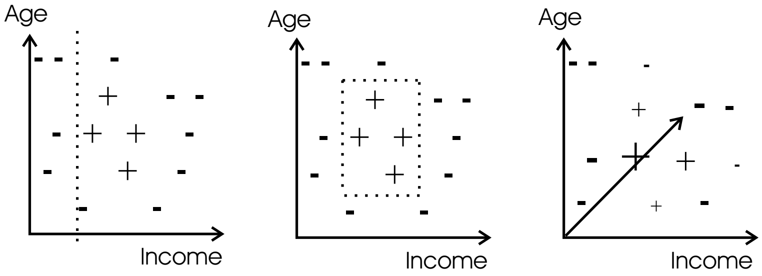

Figure 4.2: Types of virtual attributes. Left: A binary virtual attribute divides the instance space into instances satisfying or not satisfying it. Middle: A conjunction of binary attributes. Right: The most general type of virtual attributes is to add a dimension to the instance space. (The instances may for example denote credit-worthiness.)

Every virtual attribute forms an additional dimension in the instance space (see figure 4.2 (right)). This is most intuitive for numerical attributes. An example is the function

$$expectedIncomeTillRetirement(c) = (65 - age(c)) \cdot income(c)$$

Unfortunately, these dimensions can change assumptions about instance distributions and are most likely not orthogonal to the other dimensions, since they are inferrable from other attributes.

Binary virtual attributes can be visualized as separating lines within the original instance space (see figure 4.2 (left)). They divide the instance space into two regions. For example,

$$taxFree(c) \leftarrow income(c) < 330$$

may divide some instance space into salaries that are or are not subject to paying taxes in Germany. We will show that especially those virtual attributes are useful that describe target concept boundaries.

We propose to use intermediate concepts of domain theories as virtual attributes. Virtual attributes can easily be added to the set of attributes of each instance.

Intermediate attributes that are fully defined (i. e. that do not have gaps at the bottom of the domain theory) can be computed from the values of observables and other intermediates. In order to use an intermediate as a virtual attribute, it is added to the local similarities of the similarity measure. Thus, if the standard similarity measure is defined on the attribute set $\mathbb{A}$ with $|\mathbb{A}| = n$, a virtual attribute can be added as A_{n+1} as follows:

$$s(c_n, c_m) = \sqrt{\frac{1}{n+1} \cdot \sum_{i=1}^{n+1} (w_i \cdot d_i(c_n, c_m))}$$

with the local similarity for the virtual attribute A_{n+1} defined as usual. That is:
if $|A_{n+1}| \neq \infty$,

$$d_{n+1}(c_n, c_m) = \begin{cases} 1 & : \quad iff\ A_{n+1}(c_n) = A_{n+1}(c_m) \\ 0 & : \quad else \end{cases}$$

if $|A_{n+1}| = \infty$,

$$d_{n+1}(c_n, c_m) = 1 - \left(\frac{|A_{n+1}(c_n) - A_{n+1}(c_m)|}{range(A_{n+1})} \right)^2$$

Transforming domain knowledge into virtual attributes is the most general method for incorporating knowledge types that we propose in this thesis. How virtual attributes are specified will be described in the next sections which introduce subtypes of knowledge. Most of these subtypes have additional incorporation methods.
We distinguish distributional knowledge from knowledge that can be represented by virtual attributes. Distributional knowledge describes the range, density distribution and scaling of attributes. Incorporating such knowledge is common in CBR, for example attributes are often normalized by their range. Since distributional knowledge is already well researched, we focus on the less researched types of knowledge that can be incorporated using virtual attributes.

4.2.2 Matching knowledge

A common conceptualization of similarity is the amount of mutual features (Tversky, 1977; Medin et al., 1993; Goodman, 1972). Thus, one important aspect is to check whether two features are identical or match. This can be difficult due to terminology, if the case representation allows the same property to be specified in different ways. Such a problem is prone in predicate

logic representations (Bergmann, 2002), where a property might be specified with an abstract predicate (e. g. $mother(x)$ to denote that x is a mother) or with more primitive predicates ($female(x)$ and $parent(x)$). To overcome such differences, domain knowledge is needed. We refer to such knowledge as matching knowledge. Basically, matching knowledge is used to match feature values that are syntactically different but semantically equivalent. Nominal matching knowledge states that two values of an attribute are equivalent. Numerical matching knowledge defines regions in the instance space in which cases are believed to be classified identically.
An examples for numerical matching knowledge is the region specified by $pressure(c) > 30 \ \wedge \ pressure(c) < 50$ in an instance-space spanned by the numerical attributes *pressure* and *temperature*.
Nominal matching knowledge is for example the statement $poor \equiv very\_poor$, where $poor, very\_poor \in speech$ specify the speech ability of a patient (Porter et al., 1990).

Definition 4.2.1. *Two attribute values* $A_i(c_m) = a_m, A_i(c_n) = a_n \in A_i$ *are equivalent, iff* $\forall c_m, c_n : A_i(c_m) = A_i(c_n) \rightarrow A_t(c_m) = A_t(c_n)$. *We write* $a_m \equiv a_n$.

In other words, two attribute values are equivalent if replacing one value with the other does not change the class membership of the case.

Definition 4.2.2. Nominal matching knowledge *is of the form* $a_i \in A \equiv a_j \in A, i \neq j$.

Definition 4.2.3. Numerical matching knowledge *defines an interval* $R \subset A$ *on a numerical attribute* A.

Nominal matching knowledge can be incorporated into a similarity measure by widening the constraint for $d_i(c_n, c_m) = 1$ in a local similarity function as follows: Instead of
if $|A_i| \neq \infty$,

$$d_i(c_n, c_m) = \begin{cases} 1 & : \ iff \ A_i(c_n) = A_i(c_m) \\ 0 & : \ else \end{cases}$$

we use
if $|A_i| \neq \infty$,

$$d_i(c_n, c_m) = \begin{cases} 1 & : \ iff \ A_i(c_n) \equiv A_i(c_m) \\ 0 & : \ else \end{cases}$$

In other words, the local similarity for an attribute is maximal not only if the attribute values are identical, but also if they are equivalent.

Numerical matching knowledge can be incorporated as follows: Instead of if $|A_i| = \infty$,

$$d_i(c_n, c_m) = 1 - \left(\frac{|A_i(c_n) - A_i(c_m)|}{range(A_i)}\right)^2$$

we use
if $|A_i| = \infty$,

$$d_i(c_n, c_m) = \begin{cases} 1 & : \; iff \; A_i(c_n) \in R \wedge A_i(c_m) \in R \\ 1 - \left(\frac{|A_i(c_n) - A_i(c_m)|}{range(A_i)}\right)^2 & : \; else \end{cases}$$

for a region $R \subset A_i$. If two attribute values are both within the region R, their similarity is maximal.

Furthermore, matching knowledge can be incorporated using virtual attributes. $A_u(c) \leftarrow A_i(c) \in R$ with $R \subset A_i$ is a virtual attribute for numerical matching knowledge. Nominal matching knowledge of the form $a_i \in A \equiv a_j \in A, i \neq j$ can be transformed into a virtual attribute $A_v(c) \leftarrow A(c) = a_i \vee A(c) = a_j$.

For numerical and nominal matching knowledge there exist at least these two incorporation methods. As we will see later they have different behavior if the knowledge is inaccurate and have different impact on classification accuracy. Additionally, using a virtual attribute requires adjusting an additional parameter, namely the virtual attribute's weight. The other incorporation method is parameter-less.

Let us now examine where matching knowledge has been used.

A special instance of matching knowledge are taxonomies. Symbolic attribute values form nodes, and subclass- and instance-relations link the nodes. Bergmann (Bergmann, 2002) handles taxonomies as n-ary trees. Inner nodes correspond to abstract attribute values and leaf nodes to primitive leaf nodes. For example, in the domain of product recommendations for graphic cards (Bergmann, 1998), an abstract value would be *S3 Virge Card* and a primitive value would be *ELSA 2000*, stating that the latter is a subclass of the former (see figure 4.3). Cases in the case-base are represented by primitive values only, but queries can be formulated using primitive and abstract values.

According to Bergmann, taxonomies include two kinds of knowledge: Knowledge that certain classes of objects exist in the domain, and knowledge about the similarity between leaf nodes.

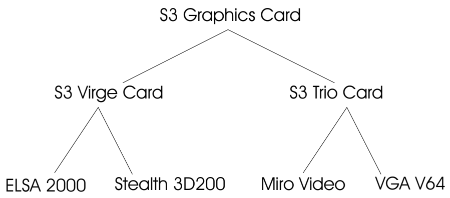

Figure 4.3: Part of a taxonomy for graphics cards. Adapted from Bergmann, 1998.

It is obvious that taxonomies are an instance of matching knowledge, since they specify how syntactically different attribute values can be matched. For example, in the graphic cards domain, the user may issue a query for a *Stealth 3D200* card. If the shop does not have that specific brand, it can offer a *ELSA 2000* card as alternative, because both cards are instances of the *S3 Virge Card* class. In this sense, the system assumes $ELSA2000 \equiv Stealth3D200$, where $ELSA2000, Stealth3D200 \in GraphicCards$.

The same methodology has been proposed in the design of aeroplanes, where less-detailed parts such as "engine" are matched to high-detailed parts such as "tail engine" (Leake & Wilson, 1999).

In the famous PROTOS system (Porter et al., 1990) explanations are used to match attribute values to one another by a number of domain-independent relations. Such relations are for example *often-correlates-with*, *causes*, or *is-a*. For example, assume that in a car-diagnosis domain two cases differ only in that one has the feature *hard-driven* and the other has *broken(carburretor)*. Given the relation *causes(hard-driven, broken(carburretor))*, the two cases can be matched as being equivalent. Similar relations are used in (Aamodt, 1994) and form inference chains. An example for such an inference chain is $lowVoltage(battery1)$; $isSubClassOf(lowVoltage(battery1), electricalFault)$; $causes(electricalFault, motorDoesNotStart)$, which is an explanation for why the motor does not start if the battery is low. If two cases are present that differ mainly in that one has the feature $lowVoltage(battery1)$ and the other has the feature $motorDoesNotStart$, their differences can be "explained away", so that they are judged similar.

In other systems, matching knowledge is used along with other types of knowledge. For example, in (Rodriguez, 2001) a taxonomy is combined with what we call causal knowledge (see section 4.2.3).

While the above approaches assume nominal attributes, matching knowledge can also be used for numerical attributes. By defining intervals on the attributes, it can be stated that the behavior or classification does not change within the interval. In simulated soccer for example, the behavior of the goalie is strongly dependent on whether it is in the penalty area or not, because it has privileges within that area. Defining the penalty area corresponds to defining intervals on the numerical position attributes (cf. (Steffens, 2005d)). Such a procedure is in many respects similar to approaches of qualitative representation (Dylla et al., 2005). While there is no widely accepted definition for "qualitative representation", such approaches usually discretize numerical attributes by defining intervals in which changes of the value are not important. For example, instead of representing the direction of an agent accurately, it might be roughly represented as one of eight partitions of the 360 degree circle (Dylla et al., 2005). Work on qualitative representations suggests that well-chosen intervals are often adequate for describing situations which traditionally were represented on a numerical scale.

Matching knowledge is the principle that underlies the approach of using generalized cases. A generalized case is a set of cases that have identical or similar classifications. This set can be represented extensionally (Bergmann, Vollrath, & Wahlmann, 1999). But generalized cases can also be represented by constraints that specify intervals (Tartakovski & Maximini, 2003). In both approaches the idea of generalized cases is to define subregions in the instance space where the classification does not change. Bergmann noted that taxonomies are strongly related to generalized cases, because an abstract node can subsume all cases of a generalized case (Bergmann, Vollrath, & Wahlmann, 1999). This observation fits nicely into our framework, because both taxonomies and generalized cases are special kinds of matching knowledge.

In summary, matching knowledge is obviously an important aspect when determining the similarity of two objects or cases and can be found in many CBR systems that employ background knowledge.

4.2.3 Inferential knowledge

Causal models have been proposed to represent the necessary information for explaining classifications. For example, in Koton's CBR system CASEY (Koton, 1988) (cf. (Aamodt, 1990)) cases contain explanations relating attributes to the classification goal using *cause* relations. Attributes can be identified as relevant for the classification if they are roots of explanations. Attributes that never appear in explanations are deemed irrelevant and excluded from the comparison process of cases. A similar approach has been described by Cain, Pazzani and Silverstein (Cain et al., 1991). The difference in their approach is that the relevance of a feature is not represented binary, but can be adjusted by a parameter which determines the influence of explanations on the similarity assessment.

We term this type of knowledge inferential knowledge, as it defines intermediate attributes on the basis of observables using logical inference. In other words, the value of an attribute can be inferred from other attributes.

Definition 4.2.4. Inferential knowledge *is a non-empty set V of rules (see definition 3.5.4 for the definition of a rule).*

Each rule $\rho \in V$ can be incorporated into a similarity measure as a virtual attribute, as described in section 4.2.1. The difference to the type "virtual attributes" is that the latter comprises all types that can be incorporated as virtual attributes, whereas inferential knowledge is defined by its representation in rule form. For example, matching knowledge can - as discussed above - be incorporated as virtual attributes, but it is not an instantiation of inferential knowledge, because matching knowledge per se does not explicitly define new attributes.

Note that we do not distinguish between inferential and causal knowledge on this level, although strictly speaking there is a difference. For the purposes of incorporating knowledge into similarity measures, modelling causality equivalently to inference will be sufficient. Inference is weaker than causality, thus, if causality between two features is given, inference from one to the other feature can be assumed: If A causes B, then it can be inferred that an object that has feature A will also have feature B.

Interestingly, the term "explanation" is used in these approaches, just as in approaches that used matching knowledge. In fact, the notion of explanation is most prominent in the area of explanation-based CBR (EBCBR). The idea of EBCBR is to use inference chains (i. e., explanations) in order to justify the classification of a case, thereby determining or influencing the similarity between new and stored case (Aamodt, 1994; Bergmann et al., 1994).

On a more detailed level, the notion of explanation is rather diverse. As described in the previous section, explanations can be used to match values that are syntactically different. In this sense, they are matching knowledge. In contrast, the purpose of Koton's and Cain's explanations was to filter irrelevant attributes. And yet another notion of explanation is proposed by Bergmann et al. (Bergmann et al., 1994) for problem-solving tasks. Here explanations are explicitly stored with the cases and are proofs of the correctness of a solution. In the approaches on matching knowledge, the explanations were generated during the case retrieval and were not permanently stored. More importantly, the way explanations influence similarity is different. In PROTOS, explanations are used to influence the similarity between cases by matching attribute values. In contrast, Bergmann defines similarity between explanations themselves, so that cases are regarded as similar if they have similar explanations.

The representation of explanations remains basically the same throughout these approaches. An explanation is an inference chain from observables over intermediates to classification goals. Still, these approaches are very different. How can this variety of the notion "explanation" be differentiated? We propose to differentiate approaches based on how they use the knowledge for similarity assessment. The framework which is introduced in this chapter is based on the use of the knowledge. For example, according to our categorization of knowledge types, the systems of Porter and Aamodt use matching knowledge as it increases the local similarity of a nominal attribute if there exists a relation between the values of that attribute. As opposed to this, CASEY uses inferential knowledge in order to infer the target attribute from the case attributes. Similarly, Bergmann et al. use inferential knowledge to link case attributes to the target attribute.

The difference between matching and inferential knowledge is also reflected in the domain theories used in the different approaches. Bergmann et al. use *strong* domain theories, that is, inferences are strictly logical. The domain theory used in PROTOS is *weak*, that is, relations have assigned confidence values and are more about correlations than about strict logical inference.

In this section we have shown how different approaches that were subsumed under the notion of explanation-based CBR could be differentiated by how they use knowledge. We have shown that the explanations in Aamodt's approach correspond to matching knowledge, whereas explanations in Koton's and Bergmann's systems make use of inferential knowledge.

However, since both matching and inferential knowledge can generate explanations, it is reasonable to group them under a common father node in a hierarchy of knowledge types.

Note that often it is hard to discern between inferential and matching knowledge, because inferential knowledge usually makes use of matching knowledge. Especially if nominal attribute values are to be inferred, the inference rule needs to define regions of the instance space in its condition part. Such regions are defined by matching knowledge.

4.2.4 Transformational knowledge

Since similarity measures work always on representations, one major issue in CBR is to guarantee that similar cases have similar representations. Yet, in complex domains superficial case representations often do not reflect the underlying similarities.

A typical domain where this is an issue is the architectural domain where new designs are generated by adapting previously designed CAD plans instead of creating them from scratch. Börner (Boerner, 1994) states that in this domain case-specific knowledge is not sufficient, but rule-based domain knowledge about transformations of the cases is necessary. Thus, in their system case representations are modified by transformations in order to be matchable. For example, rotated structures are identified as similar despite their original dissimilar representation in the attribute-value approach. For illustration, assume there are two cases $copy(X, 3, circle)$ and $copy(Y, 3, circle)$. The term $copy(d, n, s)$ denotes that shapes of the type s are copied n times into direction d. So the former case represents a horizontal line of three circles and the latter represents a vertical one. Knowledge about rotations of 90 degrees is represented as $rotate(copy(X, n, s)) = copy(Y, n, s)$. Using this transformation, the two cases are regarded as structurally similar [1]. Other transformations are scaling and mirroring. Note that if transformations are applied consecutively, they correspond to the relation-chains of the PROTOS system. Thus, transformational knowledge is reminiscent of matching knowledge, as it tries to match cases that are represented differently.

An example for approaches where transformation knowledge is not represented explicitly, but is coded directly into the similarity measure is (Coulon & Steffens, 1994). The method scales images in order to "view them from a distance", so that detailed differences vanish and more general similarities can be detected. From the knowledge-container perspective, the knowledge of transformation is encoded in the similarity measure.

[1] Note that in this example, also anti-unification would be suitable to compute structural similarity. However, for other transformations such as scaling the application of anti-unification is not as straight-forward and would require some rule-based theory, too

In the domain of simulated soccer we used transformational knowledge in order to match situations from one wing of the field to the other (Steffens, 2005d). This was done by exploiting the information that the playing field can be mirrored. Using a mirroring operation, a situation where a forward attacks from the left and has a defender blocking his way to the goal can be matched to a situation where the forward attacks from the right with a defender blocking his way.

Outside of CBR, the relevance of transformational knowledge for concept learning has been reported in the area of constructive induction. For example, Ragavan and Rendell showed that knowledge of a symmetry transformation leads to improved learning of tic-tac-toe concepts (Ragavan & Rendell, 1991). They showed that the worth of the symmetry transformation can be directly stated as reduction of required training instances. In a following chapter we will use the knowledge container approach to examine whether inclusion of a symmetry transformation into the similarity measure will reduce the number of required cases in the case-base container, too.

To sum up, the purpose of transformational knowledge is to match cases that are differently represented but are identical if some geometrical transformations are applied. Thus, in our hierarchy transformational knowledge is a subtype of matching knowledge. We define transformation knowledge as matching knowledge that makes use of geometric transformations:

Definition 4.2.5. Transformational knowledge *is of the form* $trans(c_1) = c_2$*, where* $c_1, c_2 \in U(\mathbb{A})$*, and* $trans : U(\mathbb{A}) \rightarrow U(\mathbb{A})$ *is a geometrically motivated transformation.*

Admittedly, whether a transformation is geometrically motivated is a semantic question with some room for interpretation. However, our definition of transformational knowledge can be easily extended if there are approaches that emphasize that their transformations are not geometrically motivated.

Since transformational knowledge is a subtype of matching knowledge, its incorporation in similarity measures is analogous. The constraint for $d_i(c_n, c_m) = 1$ in a local similarity function is widened. For nominal attributes, instead of

if $|A_i| \neq \infty$,

$$d_i(c_n, c_m) = \left\{ \begin{array}{lcl} 1 & : & iff\ A_i(c_n) = A_i(c_m) \\ 0 & : & else \end{array} \right.$$

we use

if $|A_i| \neq \infty$,

$$d_i(c_n, c_m) = \begin{cases} 1 & : \; iff\ A_i(c_n) = A_i(c_m) \vee trans(c_n) = c_m \\ 0 & : \; else \end{cases}$$

The local similarity for an attribute is maximal not only if the attribute values are identical, but also if the query's value can be transformed into the case's value. Note that the symmetry of the similarity measure depends on the symmetry properties of the transformation. For instance, if the transformation is a shrink-operation as used in (Coulon & Steffens, 1994), the symmetry property is lost.
For numerical attributes,
if $|A_i| = \infty$,

$$d_i(c_n, c_m) = 1 - \left(\frac{|A_i(c_n) - A_i(c_m)|}{range(A_i)}\right)^2$$

is replaced by
if $|A_i| = \infty$,

$$d_i(c_n, c_m) = max\left(1 - \left(\frac{|A_i(c_n) - A_i(c_m)|}{range(A_i)}\right)^2, 1 - \left(\frac{|A_i(trans(c_n)) - A_i(c_m)|}{range(A_i)}\right)^2\right)$$

The local similarity is calculated as usual, unless transforming the query's attribute value leads to a greater local similarity.
Furthermore, transformational knowledge can also be incorporated as virtual attributes. Additional attributes $A_{v,i}(c) = A_i(trans(c))$ are added to the similarity measure. However, obviously this method has the disadvantage that often the virtual attributes are irrelevant. This is the case if the untransformed attribute values are more similar to the comparison partner's values than the transformed values. We will show in experiments whether the virtual attribute method is suited for incorporating transformational knowledge (see section 5.6).

4.2.5 Gestalt and grouping principles

A psychologically motivated approach has been proposed by Schaaf in another CBR system for architectural CAD plans (Schaaf, 1994). The observation is that architects often use gestalts such as "comb" or "fish-bones" to remember similar problems and their solutions. Thus, in the CBR system plans are compared using pre-defined gestalts, which are salient shapes that are represented abstractly by spatial relations. During the case retrieval, the system tries to find mutual gestalts in the query and the cases, so that the similarity assessment takes place on the gestalt level and not on the level of single lines and objects.

While the author does not call these gestalts domain knowledge, from the knowledge container perspective taken in this thesis, the gestalt set can be seen as domain knowledge in the similarity container. The gestalts are represented explicitly in a set of abstract shapes and are not case-specific. They are abstract attributes defined on the observables.

Definition 4.2.6. *A gestalt is a concept rule* $\rho : U(\mathbb{A}) \rightarrow [true, false]$ *that is motivated by Gestalt psychology.*

Similarly to transformational knowledge, the distinction between general inferential knowledge and Gestalt knowledge is a semantical one. If the definition of "motivated by Gestalt psychology" is disputed, Gestalts can also be categorized as general inferential knowledge.

Incorporation of Gestalts into a similarity measure is analogous to incorporation of a nominal virtual attribute. If ρ is a Gestalt, then the corresponding virtual attribute is A_v with $d_v(c_1, c_2) = 1$, iff $\rho(c_1) \wedge \rho(c_2)$, and $d_v(c_1, c_2) = 0$ otherwise

Gestalts are a concept from research on psychological perception (Wertheimer, 1923). Another approach that makes use of psychological principles of perception is proposed by Knauff and Schlieder (Knauff & Schlieder, 1994). They use grouping to influence similarity in a top-down approach. Objects and structures in a CAD plan are grouped based on their spatial proximity and their similarity. This way, objects are partitioned into several groupings. The plans are then compared as arrangements of such groupings instead of arrangements of individual objects. For example, a line of five squares will be similar to a line of four circles, because both arrangements can be grouped into a sequence of identical objects (see figure 4.4). The authors call this a knowledge-intensive approach, because it uses top-down concepts like groupings that are not case-specific but defined in a general way.

Although Schaaf does not call his approach knowledge-intensive, the parallel between his work and the grouping approach is apparent. The similarity is not calculated on the basis of individual objects, but rather on more abstract levels, such as gestalts or groupings.

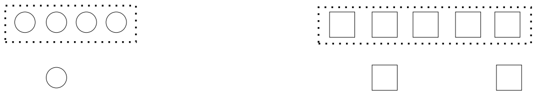

Figure 4.4: Two parts from two different CAD plans. The dotted lines mark groupings that make the two parts similar.

To place these two approaches into the canon of knowledge types, using gestalts and groupings can be seen as a specialization of inferential knowledge: Parts of CAD plans are compared to each other by means of a more general structure. Thus, two cases that are syntactically different, such as $pos(circle, 20, 10), pos(circle, 30, 10), pos(circle, 40, 10)$ and $pos(square, 50, 15), pos(square, 50, 25), pos(square, 60, 35)$ will be assigned a high similarity, because they can be represented as the same general structure, the gestalt of a line, or grouping based on proximity and similarity.

4.2.6 Contextual knowledge

Contextual attributes are only relevant in certain contexts. Thus, they are not relevant when considered in isolation, but only relevant when combined with other attributes (Turney, 1996). These other attributes specify the context in which the contextual attribute is relevant. For example, in medical diagnosis, the age of a patient is a contextual attribute, because it is uninformative when considered in isolation, but informative if combined with other attributes such as blood pressure (see (Turney, 1996) for an overview). Knowledge-light case-based approaches do not perform well in the presence of contextual attributes (Aha, 1991a).

We term knowledge that specifies in which contexts an attribute is relevant as contextual knowledge.

Definition 4.2.7. Contextual knowledge *for an attribute* A *is of the form* $relevant(A, c) \leftarrow \rho(c)$*, where* ρ *is a concept rule.*

Contextual knowledge can be incorporated into a similarity measure as follows. We assume without loss of generality that the knowledge-poor similarity measure included the attribute A_m with a global weight. In the knowledge-rich similarity measure, the attribute is only included if it is relevant for the query. Let $relevant(A_m, c) \leftarrow \rho(c)$ be the contextual knowledge to be included.

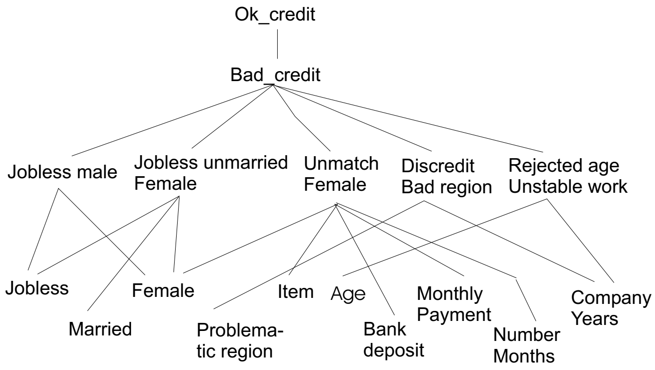

Figure 4.5: The domain theory of the JCS domain (left) and of the PGS domain (right). Nodes denote attributes and the arcs denote that the more general attribute is defined based on the more primitive ones.

$$s(c_1, c_2) = \begin{cases} \sqrt{\frac{1}{m} \cdot \left(\sum_{i=1}^{m-1} (w_i \cdot d_i(c_1, c_2)) + w_m \cdot d_m(c_1, c_2)\right)} & : \quad iff \rho(c_1) \\ \sqrt{\frac{1}{m} \cdot \sum_{i=1}^{m-1} (w_i \cdot d_i(c_1, c_2))} & : \quad else \end{cases}$$

The weight w_m can either be set to a default value or it can be learnt. In the latter case, the benefit of contextual knowledge is to speed-up weight learning if also the other weights are learnt (see section 5.5 for details).

Just as virtual attributes are defined on observables and intermediates, the inclusion of a contextual attribute depends on the value of other attributes. Contextual knowledge can be acquired from domain experts, e. g. "The position of the goalie is only relevant if the opponent's forwards have the ball" in simulated soccer. It should also be possible to infer contextual knowledge from a domain theory. As an illustrative example, consider the domain theory for Credit Screening in Japan acquired from domain experts (Blake & Merz, 1998) depicted in figure 4.5. It is apparent that the feature *married* is only relevant if the features *female* and *jobless* are present. As an approximation of relevance, one can use the following heuristic: If an attribute appears only in a subtree, then the context in which it is relevant is defined by the other attributes in the subtree.

Our notion of contextual knowledge does not cover approaches as (Jurisica, 1994) where context is given directly as set of relevant attributes. In that work, in an information retrieval system the user wants to find similar doc-

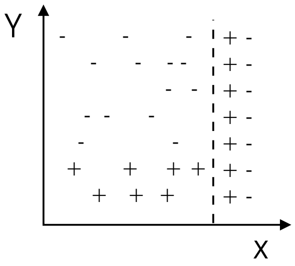

Figure 4.6: Contextual attribute. Negative instances are denoted by $-$, positive instances by $+$.

uments to a query document and explicitly specifies which attributes are relevant for his query. While this set of relevant attributes is called context, we do not view it as contextual knowledge because the relevance of an attribute is not inferred from other attributes, but is explicitly given.

In (Steffens, 2004a) we showed that contextual knowledge is useful for similarity-based opponent modelling. In simulated soccer, the attributes describing team A's defenders are irrelevant if team A's forward has the ball and is close to the opponent goal. But if team B's forward has the ball, the attributes describing team A's defenders are highly relevant. Thus, these attributes are contextual and are dependent on the context defined by which team has the ball and by the ball's position. We used a rule-base that determined in which contexts the attributes were relevant.

Contextual knowledge is a special form of inferential knowledge. It infers from other attributes whether an attribute is relevant. Thus, regions in the instance space can be defined and for each region it can be stated whether the contextual attribute is relevant or not. A simple example for contextual attributes can be seen in figure 4.6. The attributes X and Y span the instance space. The positive instances of a target class are depicted by "+", the negative instance by "-". The attributes are contextual, because in the region on the left of the imagined dotted line X is not relevant for determining class membership. X is only relevant if it is greater than a certain value. Similarly, Y is contextual because it is only relevant on the left of the dotted line, but irrelevant on the right of the line.

If a similarity measure is based on context, it is not symmetrical anymore, because the region of the query defines the context.

Contextual attributes are common in machine learning data sets, but their existence is often not acknowledged nor exploited (Turney, 1996). In this thesis we will show how contextual knowledge can be exploited in similarity measures. In the section about attribute relevance (section 4.2.7), we will review the issue of context from another perspective.

4.2.7 Attribute relevance

It has often been observed that different attributes have different relevance in determining the classification of an object. In CBR, this relevance is captured by attribute weights. The more relevant an attribute is, the higher is its weight, and the higher is its influence on similarity. Another observation is that the importance of attributes depends on the context and the goal of classification which leads to local weighting (Atkeson et al., 1997), which weights an attribute differently in different regions of the instance space. However, the most common method is to use global weighting so that attribute weights do not change across the instance space.

The relevance of attributes and attribute weights are clearly parts of domain knowledge (Richter, 1995; Gabel & Stahl, 2004). Usually, attribute weights are represented globally and statically (Wettschereck et al., 1997). That is, each attribute is assigned a weight at design time and the weight does not change. In contrast, in the afore-mentioned PROTOS system weights are dependent of the classification goal, each feature has an importance factor for each classification goal. Such class-specific weights can for example be learnt by the CBL4 algorithm proposed by Aha (Aha, 1991a). Even more dynamically, in (Clark, 1989) rules directly modify the attribute weights depending on the context.

Typically, weights are either explicitly set by domain experts (Clark, 1989) or learnt from training instances (Wettschereck et al., 1997). An intermediate approach is to infer attribute weights from the domain knowledge. In a CBR approach for software classes (Spanoudakis & Constantopoulos, 1994) the relevance of attributes is determined by their *charactericity*, *abstractness*, and *causality*, which are calculated from the domain model. Causality means that an attribute influences the value of another attribute. Charactericity is a measure for how distinct the attribute value ranges are among the different classification classes. That is, an attribute is characteristic for a class, if given the attribute value one can predict class-membership with a high certainty. Finally, abstractness is a measure for how essential the attribute is for the identity of a concept. For example, abstractness is high if the class introduced the attribute, and low if it inherited the attribute from a superclass.

In order to insert weights as knowledge type into a hierarchy, we examine its relation to contextual knowledge. Similarity measures that employ local weighting are context-sensitive (Ricci & Avesani, 1995). In such a measure the attribute weights depend on the query, that is instead of w_i, $w_i(c)$ is used as weight for attribute i. This way, context-dependent relevance can be stated, e. g. "if attribute A_1 is less than 50, set the weight of A_2 to 0". This is similar to the problem stated in section 4.2.6. The correspondence to contextual knowledge is apparent. As we have argued elsewhere (Steffens, 2005d), weights are a special form of contextual knowledge in our hierarchy. Weights express the relevance of a feature on a numerical scale. Global weighting corresponds to a context that subsumes the whole instance space. Local weighting uses contextual knowledge to specify the relevance in different regions. In other words, local weights are inferred from the attributes, just as in contextual knowledge.

Definition 4.2.8. Attribute relevance knowledge *for an attribute A is of the form $relevant(A, c, w) \leftarrow \rho(c)$, where ρ is a concept rule.*

Note that knowledge about global weights can be specified as $relevant(A, c, w) \leftarrow true$ to denote that the weight of attribute A is w.

We assume that the instance-space is partitioned into several subregions and that the attribute relevance remains the same in a subregion. We do not handle attribute weights that change numerically.

Attribute relevance knowledge can be incorporated into a similarity measure as follows. We assume without loss of generality that the knowledge-poor similarity measure included the attribute A_m with a global weight. In the knowledge-rich similarity measure, the attribute's weight depends on the context of the query. Let $relevant(A_m, c, w) \leftarrow \rho(c)$ be the attribute relevance knowledge to be included.

$$s(c_1, c_2) = \begin{cases} \sqrt{\frac{1}{m} \cdot \left(\sum_{i=1}^{m-1} (w_i \cdot d_i(c_1, c_2)) + w \cdot d_m(c_1, c_2)\right)} & : \quad iff \rho(c_1) \\ \sqrt{\frac{1}{m} \cdot \sum_{i=1}^{m-1} (w_i \cdot d_i(c_1, c_2))} & : \quad else \end{cases}$$

Typically, for each subregion there will be a rule defining the attribute's weight.

4.2.8 Distributional knowledge

A type of knowledge which is placed apart from the other types mentioned so far was proposed in (Surma, 1994). Matrices specify the ordering of nominal attributes, i. e. similarities within the value set of an attribute itself. The rows and columns denote the attribute values, and the cells contain the similarity between two attribute values. The assumption is that even in a nominal attribute, some value pairs are more similar to each other than others. For example, if a feature *color* has the possible values *yellow, orange, blue*, then *yellow* is more similar to *orange* than to *blue*. This allows to enhance matching results from the dichotomy "match" vs. "no-match" to numerical similarities. While in (Surma, 1994) the similarities between attribute values were given by a domain expert, these similarities were learned in (Baumeister et al., 2002).

Knowledge about the ordering of nominal values and the similarity between values is distributional knowledge, since it specifies the scale of an attribute. We do not give a formal definition here, because we will not discuss ordering of nominal values in the remainder of this thesis.

As described in the introduction, distributional knowledge specifies the range, density distribution and scaling of attributes. Although distributional knowledge is a type of itself, it can be combined with other types. For example, it can be combined with contextual knowledge to state information such as "If an attribute is given, other attributes have a specific range" (Gabel & Stahl, 2004).

4.2.9 Summarizing remarks on knowledge used in CBR

We reviewed kinds of knowledge that have been used in case retrieval. We have ignored approaches that used domain knowledge to adapt retrieved solutions to the new case (see section 2.5.4 for references). Adapting solutions is out of the scope of this thesis.

In the case retrieval approaches, knowledge is typically not formally defined, but used in an ad-hoc fashion. To our knowledge we presented the first systematic grouping of knowledge types for case retrieval. Moreover, domain knowledge has mostly been used for structured representations so far. In the past, it has even been claimed that attribute-value representations inhibit the incorporation of domain knowledge (Branting, 1989), but this view has been abandoned nowadays (e. g. (Aha, 1991b)). Our proposed incorporation methods have been described with attribute-value representations, and our experiments (to be reported later) show that similarity-based classification can indeed benefit from domain knowledge even with attribute-value representations.

For example, in simulated soccer we have shown that using inaccurate domain knowledge can improve the accuracy of predicting the opponent's actions (Steffens, 2004a, 2005d). In those experiments we used goal-dependency networks (GDNs) as proposed by Stepp and Michalski (Stepp & Michalski, 1986). In our framework, GDNs specify contextual and inferential knowledge. More details about the experiments can be found in chapter 7.

The main disadvantage of using additional background knowledge in CBR is that of higher computational cost. For example, matching attribute values by relation-chains as in PROTOS is more complex than using a simple identity operator. Thus, the additional effort has to be motivated by accuracy improvement or case-base reduction. In the later chapters we will analyze the effect of knowledge types on classification accuracy.

4.3 Psychological knowledge types

In this section an overview over the knowledge types that have been proposed to influence psychological similarity assessment is given. Research on similarity includes categorization, analogy, object recognition, and similarity in general. Often, in the psychological literature the researched principles are not called knowledge. Yet, as will be seen, many of them correspond to the knowledge types defined in the previous sections.

However, the technical knowledge container perspective is of no use to decide whether a psychological phenomenon is based on knowledge. Rather, Hahn (Hahn & Chater, 1998) makes a distinction between knowledge-based factors and process principles. We will follow this distinction and use the notion of knowledge conservatively.

4.3.1 Causal knowledge

Causal knowledge is believed to influence attribute relevance for similarity assessment. Features that cause others are deemed more relevant than attributes that are effected by others (Ahn et al., 2000). Additionally, causal attributes are deemed more important than correlated ones (Choplin et al., 2001). Causal background knowledge can also be used to infer missing or unobservable attributes (Ahn et al., 2000).

In contrast to research in CBR, here the focus is not on inferring additional abstract attributes, but to determine which attributes are salient. The assumption is that humans possess a large amount of information about properties of objects, and not all of this information can be used for similarity assessment. This is different from the CBR perspective, where every information has to be given explicitly. Psychological research is thus concerned with how to constrain the available information. Often, salience and relevance are mixed, so that attributes that are salient are relevant for the comparison process. In this light, the finding that causal knowledge influences attribute relevance is less similar to attribute relevance knowledge and more reminiscent of explanation-based CBR. In EBCBR irrelevant attributes were filtered if they were not used in explanation-chains from the observables to the classification goal. This is the same concern as the psychological approach to identify principles that constrain information. Thus, the causal knowledge used in psychology corresponds to inferential knowledge in our hierarchy. The distinction between causality and inference is not existent or at least not clear in the cited psychological work.

4.3.2 Contextual knowledge

Aha and Goldstone suggest that attribute weights are not represented statically, but dynamically based on the case's context (Aha & Goldstone, 1992). The context to determine an attribute's importance is defined as the set of other features that are present in the case. As an example they state that the importance of the feature *date of the next deadline* for the category *will work on the weekend* depends on the attribute *upcoming computer downtime*. That is, the importance will be greater if there is a computer downtime directly before the next deadline.

Aha and Goldstone also showed that subjects can learn context-specific weights from a set of cases. Depending on where in the instance-space the case was located, attribute importance was estimated differently. Furthermore, they developed the GCM-ISW algorithm that learns contextual weights and fits the psychological data closely.

In the experiments, subjects had to categorize stimuli into category A or B. The stimuli consisted of a square with eight possible sizes and a vertical

bar at eight possible horizontal positions. For each category there were two clusters, which were constructed in a way, so that differentiating between the first cluster of A and the first cluster of B the position of the bar was relevant. For differentiating between the second cluster of A and the second cluster of B, the size of the square was relevant (refer to figure 4.6 for a rough idea).
An example of the verbal protocol that subjects gave during classifying stimuli (Aha & Goldstone, 1992) (p. 537):

> ”I looked at the size of the square. If it was big, then I looked at where the bar was. If it was a little further to the right, then I put it in A. Otherwise I put it in B. If the square was small, I looked carefully at its size. A squares were slightly bigger than B squares.”

This can be stated in our terminology as $relevant(barPosition, c) \leftarrow size(c) > someSize$ and $relevant(size, c) \leftarrow size(c) < someSize$. It is apparent that this form of contextual knowledge fits our notion as defined in section 4.2.6. Thus, using contextual knowledge in similarity-based classification is psychologically plausible.
Interestingly, also a finding by Gentner (Medin et al., 1993) can be related to contextual knowledge, where she states that an object’s shape is considered as relevant if an object has to be classified into a category that is described by a noun. For example, if the target concept is described by an adjective such as ”eatable”, children consider the shape of an object less relevant than if the target concept is described by a fantasy noun such as ”wug”.

4.3.3 Deep attributes

Similarity can be based on so-called deep attributes if the person possesses background knowledge for the domain (Chi et al., 1981; Medin et al., 1993). Such deep attributes are distinguished from superficial attributes by the fact that they cannot be directly perceived in the objects but have to be inferred using background knowledge. For example, if the similarity between exercises in physics is to be assessed, superficial attributes are those that are given in the exercise text. Deep attributes are not given explicitly, but have to be inferred. From the superficial attributes ”collision”, ”momentum” etc. that are present in the exercise text the deep attribute ”copes with conservation of energy” must be inferred in order to identify the solution method for the exercise (Chi et al., 1981).

Obviously, superficial attributes correspond to observables in our terminology. Deep attributes correspond to the intermediate attributes that can be derived from observables using inferential knowledge.

For psychology, the possibility to use deep attributes raises the question of which knowledge is activated for similarity assessment. It is not feasible to assume that the complete knowledge of a human is used during similarity assessment (Medin et al., 1993). A proposal to remedy this issue was given by Barsalou (Barsalou, 1989). He regards concepts as being represented in different parts. The core part describes properties that are always activated if the concept is reasoned about. In another part context-specific properties are stored that are only activated in certain contexts. For example, for the concept ”France” the fact that France won the soccer world-championship in 1998 will only be activated if the context is sport, and not politics or economy. Further properties are stored in a part that is activated by recent knowledge, that is, by information that was acquired or active shortly before the reasoning.

Barsalou’s idea of context-dependent representations for deep attributes is another hint that inferential and contextual knowledge are intertwined.

4.4 Learning knowledge types

The bottleneck of knowledge-rich similarity is acquiring the domain knowledge. The additional effort of knowledge engineering must be minimized. Thus, an important aspect is to analyze the requirements for the knowledge. If imperfect knowledge will turn out to be useful, this will facilitate knowledge acquisition as it lightens some requirements such as correctness, consistency and/or completeness.

If no domain experts are available or too expensive, partial and inaccurate (or vague) knowledge can also be acquired with machine learning or statistical methods. Before employing machine learning methods it must be clear which knowledge can be incorporated into similarity measures. The above definitions of knowledge types will be useful to decide for which knowledge the cases should be processed for.

In this section we will give a short overview over which methods can be used to learn knowledge of the defined types. However, the focus of our thesis is not to learn knowledge, but to examine how different types of knowledge effect similarity-based classification and what the impact of imperfectness is. Thus, we will only give references and ideas for learning knowledge types here. Still, it will be interesting to examine how other machine learning methods can be integrated into CBR in future work.

Of course, the issue of applying other learning methods than CBR raises the question why not to learn the whole target class with other learning methods. Yet, learning the different knowledge types requires less effort than learning the whole concept. For example, while the target class may have many concept boundaries in the instance space, one can apply learning methods to derive only one of these boundaries, or to check whether some of the boundaries are axis-parallel. As we will see in the next chapter, such knowledge chunks are useful for similarity-based classification. Such knowledge can be learned with less effort than learning the whole concept.

Furthermore, learning from training data is inherently prone to inaccuracies. Thus, learning from data will always only yield an estimation of the correct information. In contrast to lazy learning approaches, many eager learning algorithms learn hypotheses in a global way and are thus sensitive to noise. In this thesis we investigate how imperfect, inaccurate knowledge influences similarity-based classification.

4.4.1 Contextual knowledge

The assumption that learning knowledge chunks is easier than learning the whole target concept is intuitively clear if one considers learning contextual knowledge, where the information that an attribute is relevant in some region can be learnt easier than the exact function that separates class-members from non-members in that region.

To illustrate this, consider decision tree learning (Quinlan, 1993). Decision trees partition the instance-space with the attributes that have the highest information-gain, that is, that best separate positive from negative instances. Attributes are selected sequentially, until no attribute can further increase the classification accuracy. Since decision trees can only use axis-parallel separation lines for partitioning, there might be concepts that cannot be learnt perfectly. In such cases, it has been proposed to use the imperfectly classifying decision tree for CBR (Ling, Parry, & Wang, 1997). The information gain of the attributes in the decision tree are used as weights in the similarity measure of CBR. Attributes that did not appear in the decision tree were

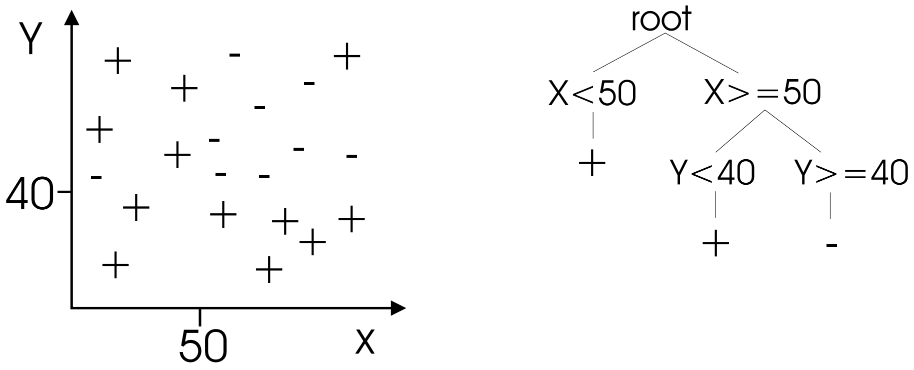

Figure 4.7: Left: A non-noisy instance space, with "+" denoting positive, and "-" denoting negative instances of some concept. Right: An imperfect decision tree describing the concept.

deemed irrelevant and weighted with 0. While up to now these weights were used globally, it is obvious, that the paths in a decision tree define a context (cf. (Domingos, 1997)). For example, in figure 4.7, it is shown that in the subtree under $X < 50$ the attribute Y is not used. This can be stated in our terminology as contextual knowledge $relevant(Y, c) \leftarrow X \geq 50$.

The tree-structure of decision trees is not equivalent to the structure of domain theories. However, the definition of context in decision trees is similar to inferring context from domain theories as discussed in section 4.2.6. Yet, we have to postpone an implementation and experiments for future work.

Note that there are several other approaches to learn context-sensitive feature weights (e. g. (Domingos, 1997; Aha & Goldstone, 1992)). However, for contextual knowledge as defined in section 4.2.6, we approximated feature relevance as binary and assume that feature relevance does not change within specific regions. Since context-sensitive feature weighting stores weights for each exemplar, the weights are not constant over regions, but may change from instance to instance. Thus, traditional context-sensitive weight learning methods do not output the information that we need for contextual knowledge.

4.4.2 Virtual attributes

Adding virtual attributes to the similarity measure is basically equivalent to re-representing the cases to better fit the classification task. Exactly the same motivation is behind constructive induction (CI). The idea behind CI is to learn additional features from cases that transform the instance-space so that the concept can be better approximated (Matheus, 1991). For example, if a concept was dispersed over the instance-space (which is the case for disjunctive concepts), additional features were learned on which the concept was not dispersed.

While our approach is top-down, CI is a bottom-up approach (Wogulis & Langley, 1989). These two directions complement each other nicely, as the focus of CI ends where our focus starts. While we analyze which types of knowledge are useful for similarity-measures and what the effects of imperfectness of knowledge are, CI investigated how to learn additional attributes. CI was basically concerned with rule-based processes, e. g. completing incomplete domain-theories (Mooney & Ourston, 1991; Fu & Buchanan, 1985) or to find additional features for rule-based reasoning (Matheus, 1991; Wogulis & Langley, 1989; Gunsch & Rendell, 1991). However, it has been shown that constructing features can also improve exemplar-based learning (Aha, 1991b). Aha's IB3-CI algorithm constructs simple features if the system retrieves a case c that has a different class than the query q. In such a situation the feature-set is increased in order to reduce the similarity between c and q. This is done by identifying features that are only present in q and not in c. By logically conjuncting these features, a new feature is constructed that is likely to better discriminate between positive and negative cases in future retrievals. The new feature is discarded if it is a specialization of a feature that is already contained in the feature-set. IB3-CI significantly outperforms other instance-based learning methods that do not construct features.

The methods of CI, such as IB3-CI, are potential candidates for learning virtual attributes and inferential knowledge. They also show that learning individual features is easier than learning the whole target class.

4.4.3 Weights

Weights are a type of knowledge that is unlikely to be acquired from domain experts, as the semantics of numerical weights can only be implicitly defined by comparing them to other weights.

The influence of weights on classification has been studied extensively already. There exist many learning methods that estimate attribute weights from data (see (Wettschereck et al., 1997) for an overview). These weight learning methods can be organized and dichotomized in a framework introduced by Wettschereck, Aha and Mohri (Wettschereck et al., 1997). Discriminating dimensions in this framework are the following:

- Performance vs. preset bias: Weight learning methods with a performance bias analyze the result of the system's classification and adapt weights accordingly. If for example a misclassification occurred, the weights of matching attributes can be reduced and weights of mismatching attributes can be increased (Salzberg, 1991). In contrast, methods with a preset bias estimate weights by statistical or information-theoretic means. For example, the information-gain used in decision-tree learning (Quinlan, 1993) can also be used as attribute weights in similarity measures (Daelemans & Bosch, 1992).

- Continuous vs. binary weight space: In a binary weight space, weight learning is equivalent to feature selection, that is, some features are weighted with 1, and those that are weighted with 0 have no influence at all. This works well if attributes are either relevant or irrelevant. But if attributes are partially relevant, a continuous weight space has to be searched (Wettschereck et al., 1997).

- Given vs. transformational representation: Weight learning methods can either work with the given representation only, or they can change the representation. The afore-mentioned algorithm IB3-CI is an example for a method that changes the representation by adding constructed features (Aha, 1991b).

- Global vs. local generality: Weights can either be learnt globally so that they are the same over the whole instance space, or they can be learnt locally. In the latter case, the weights depend on where in the instance space the query is located.

- Poor vs. intensive use of knowledge: Another dimension is the amount of domain knowledge that is used for learning weights. An example for knowledge-intensive methods is the EBCBR approach where feature weights are set to 1 if they appear in an explanation, and to 0 else.

We implemented several of the existing weight-learning methods to learn weights for virtual attributes. More details about this are presented in chapter 6.

Recently, an approach to learn weights has been proposed for knowledge-intensive CBR (Stahl, 2004). The system adapts weights during daily usage based on utility feedback (this process is also called introspective learning (Bonzano, Cunningham, & Smyth, 1997)). To do this, the traditional CBR cycle of retrieve, reuse, revise, and retain is refined. Particularly, in the revise phase, the performance of the retrieval process is measured, and in the retain phase the similarity measure is optimized based on the performance feedback. For example, this feedback can form an error function, so that the system can update weights in a gradient descent fashion. Additionally, it was shown that genetic algorithms can learn feature weights and local similarity functions by using the error function as fitness function (Stahl, 2004).

4.4.4 Matching knowledge

Discretisation is an approach that transforms a numerical attribute into a set of nominal features (Fayyad & Irani, 1993). It outputs intervals as used by numerical matching knowledge. Surprisingly, discretisation is rarely used in lazy learning approaches, because it discards information (Ting, 1997). However, it was shown that discretisation can improve performance if the case-base is noisy or if there are many irrelevant attributes (Ting, 1997). In contrast, we will show in our experiments that defining intervals can improve classification accuracy even if the data is not noisy and attributes are relevant. Furthermore, in our approach not the whole range of an attribute is discretized, but only parts of it.

Matching knowledge is similar to the "close-interval" operator in constructive induction (Gunsch & Rendell, 1991). For example, if objects from a class are described as weighting 60, 64, and 70 ounces, and all the objects from the other classes weight less than 60 or more than 70, a new feature "weights 60 to 70 ounces" can be created.

As mentioned in the previous section, genetic algorithms have been used to learn local similarity functions (Stahl, 2004). As a special case, these algorithms could also be used to learn equivalences within numerical intervals or between nominal feature values.

4.5 Conclusion

In this chapter we have introduced a framework for knowledge types that have been used or researched in connection with similarity in cognitive science. We have cleared up the terminology by giving definitions for the knowledge types. Differences and commonalities have been discussed, so that parallels between similarity in CBR and psychology were identified. With this hierarchy we have systematized and formalized incorporation of knowledge into similarity measures, which has been previously usually done in an application-specific ad-hoc fashion.

Furthermore, we proposed several novel incorporation methods. In fact, integrating the types into a hierarchy was only possible by introducing new incorporation methods. Different types that can be used in the same way were arranged under the same more general type.

The proposed knowledge types are applicable to attribute-value representations.

In the subsequent chapters we will examine the impact of the knowledge types on classification accuracy, and we will analyze how robust the incorporation methods are if the knowledge is imperfect.

Chapter 5

Impact of Imperfect Knowledge on Classification Accuracy

> An investment in knowledge
> always pays the best interest.
>
> *Benjamin Franklin*

5.1 Introduction

Incorporating domain knowledge into similarity measures is an additional effort which has to be motivated by increased classification accuracy. In this chapter we analyze the effect of the different knowledge types on accuracy partly formally and mainly by experiments. The experiments are done in artificial and in real-world domains. Furthermore we examine how robust the incorporation methods are if the domain knowledge is inaccurate, partial, or inconsistent.

5.2 Virtual attributes

Virtual attributes are intermediate attributes that are derivable from observables. That is, only intermediate attributes that are fully defined (without gaps in the bottom of the domain theory (Mooney & Ourston, 1991)) can

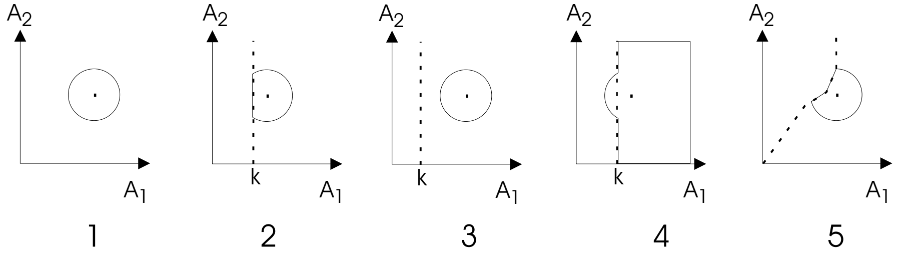

Figure 5.1: Areas of equal similarity. 1. For the standard similarity based on Euclidean distance. 2. With a binary virtual attribute $A_v(c) \leftarrow A_1(c) > k$. 3. With a binary virtual attribute but the query is far away from the separating line. 4. With a binary virtual attribute, the similarity value of the iso-area is smaller than $1 - w_v$. 5. With a binary virtual attribute that is not axis-parallel.

be used as virtual attributes. Our definition of intermediate allows a virtual attribute to be nominal or numerical (refer back to figure 4.2). Virtual attributes with a nominal domain can be seen as partitioning the instance space into regions with equal values for the virtual attribute. An example for a nominal (more specifically, binary) virtual attribute is $A_v(c) \leftarrow A_1(c) < 30$ which defines a separating hyperplane through the instance space at $A_1 = 30$. A numerical virtual attribute would be $A_v(c) = A_1(c) + A_2(c)$ which forms an additional numerical dimension in the instance space.

The effect of virtual attributes can be intuitively understood by looking at how they change the area of equal similarity around a query. If the standard Euclidean distance is used as basis for the local similarities, the cases with equal similarity to a query are located on a circle around the query (see figure 5.1). If a binary virtual attribute is added, the area of equal similarity is changed as follows: The circle of equal similarity is cut off by the hyperplane described by the virtual attribute. That is, two instances that are on different sides of the hyperplane will be less similar to each other than two instances that are on the same side. To be precise the iso-similarity area is only cut off by a virtual attribute A_v for similarities that are greater than $r = 1 - w_v$ (note that 1 is the maximal similarity, which is equivalent to identity). For iso-similarity areas with similarity values smaller than r it is not necessary anymore that the instances are on the same side of the separating hyperplane. Then the area of equal similarity will begin to grow a smaller circle around

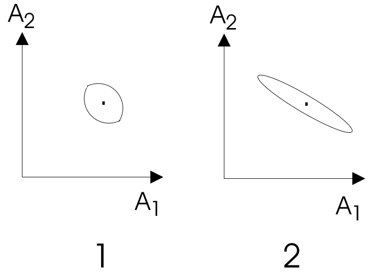

Figure 5.2: Areas of equal similarity. 1. With a numerical virtual attribute $A_v(c) = A_1(c) + A_2(c)$. 2. With the same virtual attribute, but a higher weight w_v.

the query that goes over the separating line. In a classification framework this situation is only relevant if no case has the same value for the virtual attribute as the query.

The effect of a numerical virtual attribute is different. Let us look at the universe spanned by two numerical observables A_1, A_2. If a numerical virtual attribute $A_3(c) = A_1(c) + A_2(c)$ is added to the similarity measure, the area of equal similarity (if plotted in the universe spanned by the observables) changes to a diagonal lense (see figure 5.2). By changing the weight of the virtual attribute, the lense can be thickened or thinned. The diagonal lense shape is due to the fact that the area of equal similarity for the virtual attribute alone is a diagonal stripe, i. e. low A_2 values have to be compensated by high A_1 values to achieve the same A_3 value, and vice versa.

The main difference between nominal and numerical virtual attributes is the following. For nominal virtual attributes, the area of equal similarity for a given similarity value remains the same if the query is far away from the hyperplane. The area is only affected if it intersects a hyperplane. For numerical virtual attributes, the shape of iso-similarity is inherently changed. In the following we focus on nominal virtual attributes.

5.2.1 Inaccurate knowledge

Vague knowledge about a separating hyperplane is that its position is not exactly known and only a possibly inaccurate estimation can be given.

First we illustrate the effect of adding an inaccurate virtual attribute $A_v(c) \leftarrow A_1(c) < k, k \in \mathbb{R}$ to the standard similarity measure in a universe spanned

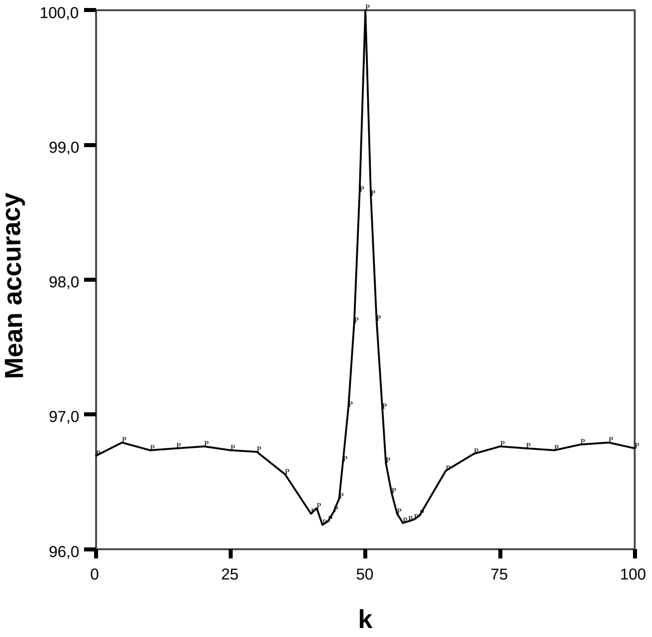

Figure 5.3: Accuracy of a similarity measure with the virtual attribute $A_v(c) \leftarrow A_1(c) > k$ for the concept $f(c) \leftarrow A_1(c) > 50$ in a two-dimensional universe. 100 cases were in the case-base, 200 test cases were used, and the experiment was repeated 3000 times.

by the attributes A_1, A_2. For illustration we choose the simple target class $f(c) \leftarrow A_1(c) > 50$. By varying k we operationalize the inaccuracy of the knowledge about the separating hyperplane. In this experiment, A_1 and A_2 are numerical in the interval $[0, 100]$, 100 random cases were in the case-base, 200 random test cases were used, and the experiment was repeated 3000 times. The mean accuracies for various values of k are depicted in figure 5.3. The similarity measure that does not use any knowledge about the separating hyperplane at all is equivalent to setting k to 0 or 100. In the experimental results, knowing the exact position of the hyperplane boosts the classification accuracy to nearly 100 %. However, the accuracy degrades rapidly if the hyperplane is vaguely approximated by k. If k is only 4% off the correct value, the virtual attributes has no positive effect anymore. Even worse, for certain

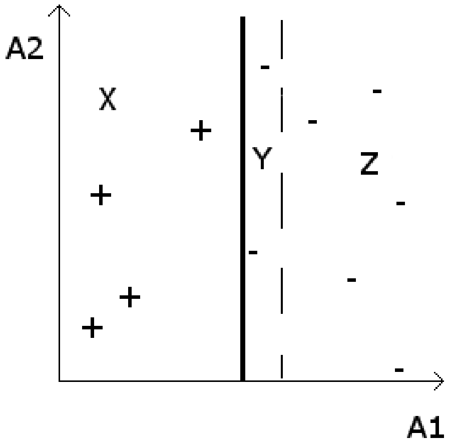

Figure 5.4: Three regions (X, Y, Z) defined by the concept boundary (solid line) and the separating line described by an inaccurate virtual attribute (dashed line).

values of k, the knowledge-rich similarity measure yields worse accuracy than the similarity measure without the virtual attribute. Surprisingly, if k is very wrong, the accuracy is as good as the knowledge-poor baseline.

Intuitive analysis:

Let us look at the effect of this virtual attribute more closely in order to understand in which situations the accuracy is increased and when it is decreased. If the virtual attribute describes the concept boundary correctly, the misclassifications at the concept boundary vanish, because the area of equal similarity is cut off at the concept boundary. If the iso-similarity area around a query is gradually increased until it includes a case from the case-base, it is guaranteed that the area only increases on the correct side of the concept boundary (unless there is no instance at all on the same side, but then a misclassification will occur with any similarity measure).
If the virtual attribute describes a separating line slightly shifted from the concept boundary, for a query three situations can occur:

1. If the query is in region Z (see figure 5.4), the estimated separating line is between the query and the concept boundary. This is a good

situation, because the query will be most similar to a case on the same side of the concept boundary (in very extreme situations no case with the same class might be on the same side of the virtual attribute, but this will be very rare and will yield the same results as the standard measure).

2. If the query is in region X, the separating line is on the other side of the concept boundary. In this situation, a misclassification is possible, if the increasing iso-similarity area reaches a case between the separating line and the concept boundary before it reaches a case with the same class as the query. The probability of a misclassification increases with the space between the separating line and the concept boundary and inversely with the distance of the query to the concept boundary. However, such a misclassification would also occur without the virtual attribute. Even more so, if the space between separating line and concept boundary is small enough so that no case is in it, some misclassifications can be avoided that would occur with the standard similarity measure. In such a situation the separating line discriminates correctly between the negative and the positive cases in the case-base, because then cases which contradict the separating line do not exist in the case-base. This is reflected by the data points in figure 5.3 that describe wrong separating lines very close to the correct value but are still above the baseline of the standard similarity measure.

3. If the query is in region Y, additional misclassifications (as compared to the standard similarity measure) can occur, because the query is between the separating line and the concept boundary. In this situation, the increasing iso-similarity area can only hit same-classed cases within the area between the separating line and the concept boundary. If this area is small (relative to the case density), it is likely that the iso-similarity area will hit a case only after it crossed the concept boundary. Then a misclassification will occur. Interestingly, the probability of misclassifications decreases if the space between the separating line and the concept boundary increases. This explains why in figure 5.3 the accuracy of similarity measures with very wrong virtual attributes is not different from the standard similarity measure.

Obviously, in the second and third situation, the probability of misclassifications depends on the density of the cases in the case-base. For example, for low densities the probability for avoiding misclassifications in situation 2 increases, but the probability of additional misclassifications in situation 3 also increases.

In figure 5.5, we compare the accuracy of inaccurate virtual attributes for different sizes of the case-base. The curve for a case-base with 100 cases (CB100) is the same as in figure 5.3, additionally there are curves for case-base sizes of 20 (CB20) and 500 cases (CB500). Apparently, the robustness for slightly wrong virtual attributes is greater for low case densities. While for CB100 the accuracy is only better than the baseline [1] if the virtual attribute is about 3% wrong, for CB20 the accuracy is better even if the virtual attribute is about 6% wrong. For CB500, the virtual attribute has to be less than 1.5% wrong. On the other hand, the probability for additional misclassifications in situation 3 is increased for low case densities which is shown by the fact that the difference between the baseline and the lowest accuracy is biggest for CB20 and lowest for CB500.

Formal analysis:

Formally estimating the effect of a binary virtual attribute in a similarity measure is difficult, since even predicting the classification error of a standard similarity measure at a vertical concept boundary in a two-dimensional space has been shown to result in a non-closed formula by Ling and Wang (Ling & Wang, 1997), so that is has to be approximated numerically. We adapt Ling and Wang's method to deal with additional binary virtual attributes. To approximate the misclassification of a similarity measure with an inaccurate virtual attribute, we consider again the three regions depicted in figure 5.4. Region Z is the area partitioned by the virtual attribute which does not include the concept boundary. Region Y is between the concept boundary and the separating line introduced by the inaccurate virtual attribute. Region X is the area of the concept that does not include the separating line.

For the formal analysis, we assume that the universe is spanned by two numerical attributes, that cases are uniformly distributed with density λ, that positive and negative cases are separated by an axis-parallel concept boundary at $A_1 = k$, and that all attributes in the similarity measure are weighted equally. The analysis is for a given d_1, which specifies the distance between

[1] The baseline is the similarity measure without a virtual attribute, which is equivalent to setting k to 0 or 100.

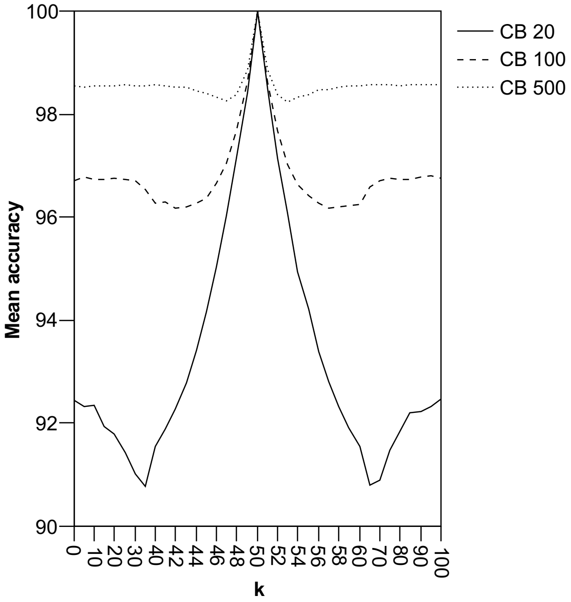

Figure 5.5: Accuracy of a similarity measure with the virtual attribute $A_v(c) \leftarrow A_1(c) > k$ for the concept $f(c) \leftarrow A_1(c) > 50$ in a two-dimensional universe for different case-base sizes. 20, 100 or 500 cases were in the case-base, 200 test cases were used, and the setting was repeated 3000 times. Note that the horizontal axis is stretched in the interval 40-60.

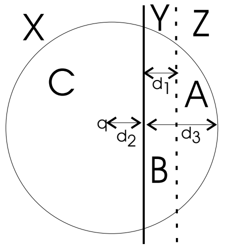

Figure 5.6: Area of equal similarity around a query q in region X. The concept boundary is depicted by the solid vertical line. The separating line estimated by the virtual attribute is depicted by the dashed vertical line.

the concept boundary and the separating line described by the virtual attribute.

To estimate the number of misclassifications, we consider queries differently depending on the region they are in. We approximate that queries in region Z are classified correctly, since the area of equal similarity is cut off by the virtual attribute before it goes over the concept boundary.

Misclassifications for queries in region X can be estimated as follows. A misclassification for a query in X can only occur if the most similar case to the query is in region B (see figure 5.6). Let p_1 be the probability that at least one case is in region B. Let p_2 be the probability that no case is in region C. Let S be the size of the universe.

$$p_1 = 1 - \left(1 - \frac{B}{S}\right)^{S \cdot \lambda}$$

$$p_2 = \left(1 - \frac{C}{S}\right)^{S \cdot \lambda}$$

Then, the error rate at the query is

$$E_q = \int_0^{100-k} p_1 \cdot p_2 \; dd_3.$$

The error introduced by a boundary with the length 1 is

$$E_X = \int_0^k E_q \; dd_2.$$

Now, the areas of A, B, C can be calculated as follows:

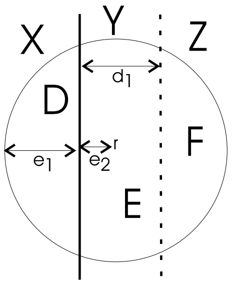

Figure 5.7: Area of equal similarity around a query r in region Y. The concept boundary is depicted by the solid vertical line. The separating line estimated by the virtual attribute is depicted by the dashed vertical line.

$$C = \pi \cdot (d_2 + d_3)^2 - (A + B)$$

$$A = (d_2 + d_3)^2 ArcCos\left(\frac{d_1 + d_2}{d_2 + d_3}\right) - (d_1 + d_2) \cdot \sqrt{(d_2 + d_3)^2 - (d_1 + d_2)^2}$$

$$B = (d_2 + d_3)^2 ArcCos\left(\frac{d_2}{d_2 + d_3}\right) - d_2 \cdot \sqrt{((d_2 + d_3)^2 - d_2^2)} - A$$

The misclassifications for queries in region Y can be estimated in a similar way. Let p_3 be the probability that at least one case is in region D (see figure 5.7). Let p_4 be the probability that no case is in region E.

$$p_3 = 1 - \left(1 - \frac{D}{S}\right)^{S \cdot \lambda}$$

$$p_4 = \left(1 - \frac{E}{S}\right)^{S \cdot \lambda}$$

Then, the error rate at the query is

$$E_r = \int_0^k p_3 \cdot p_4 \; de_1.$$

The error introduced by a boundary with the length 1 is

$$E_Y = \int_0^{d1} E_r \; de_2.$$

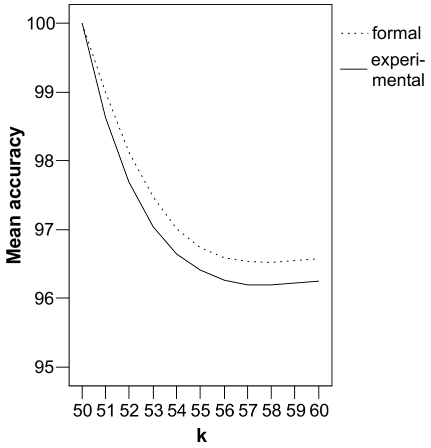

Figure 5.8: Experimental and formal accuracy of a similarity measure with the virtual attribute $A_v(c) \leftarrow A_1(c) > k$ for the concept $f(c) \leftarrow A_1(c) > 50$ in a two-dimensional universe. 100 cases were in the case-base (that is, $\lambda = 0.01$). For the experimental data 200 test cases were used, and the setting was repeated 3000 times.

The areas of D, E, F can be calculated as follows:

$$D = (e_1 + e_2)^2 \cdot ArcCos\left(\frac{e_2}{e_1 + e_2}\right) - e_2 \cdot \sqrt{(e_1 + e_2)^2 - e_2^2}$$

$$F = (e_1 + e_2)^2 \cdot ArcCos\left(\frac{d_1 - e_2}{e_1 + e_2}\right) - (d_1 - e_2) \cdot \sqrt{(e_1 + e_2)^2 - (d_1 - e_2)^2}$$

$$E = \pi \cdot (e_1 + e_2)^2 - (D + F)$$

E_X and E_Y are weighted with the probability that the query is in them:

$$E_{total} = \frac{X}{S} \cdot E_X + \frac{Y}{S} \cdot E_Y$$

In figure 5.8 we compare the analytical results with the empirical results. The curve from figure 5.3 is replotted in the interesting interval around the correct value for k. The curve for the analytical E_{total} (plotted as Accuracy=1$-E_{total}$) fits closely.

Both the formal and the experimental results suggest that similarity measures can benefit from virtual attributes describing separating lines, even if the knowledge is slightly inaccurate.

Note that this analysis can be easily extended to settings in which the two dimensions are weighted differently. In such a case, only the area calculations have to be changed from circle segments to ellipse segments (cf. (Ling & Wang, 1997)). Furthermore, the same formula holds for horizontal lines (cf. (Ling & Wang, 1997)), so that misclassification rates for concepts that are formed by axis-parallel boundaries can be calculated. We are investigating the effect of virtual attributes on concepts with several boundaries in the section on partial knowledge (section 5.2.3). For polygonal concepts whose boundaries are not axis-parallel, the equations need to be modified to depend on the angle α between the boundary and the horizontal axis (as proposed by Ling and Wang (Ling & Wang, 1997) for standard similarity measures).

5.2.2 Inconsistent knowledge

Inconsistent knowledge means that different rules make different predictions about the same fact. To evaluate the effect of inconsistent virtual attributes on classification accuracy, we set up the following experiment: Again, we have a universe spanned by two numerical attributes in the interval [0,100]. Positive and negative cases are separated by a simple vertical line at $A_1 = 50$. We incorporate two virtual attributes in the similarity measure, which both estimate the same concept boundary. By this we simulate the situation that the knowledge about the concept boundary is inconsistent. The virtual attributes are $A_v(c) \leftarrow A_1 > k_1$ and $A_w(c) \leftarrow A_1 > k_2$. By varying k_1 and k_2 we can operationalize various degrees of inconsistency. For example, if $k_1 = k_2$, there is no inconsistency. If $|k_1 - k_2|$ is small, the inconsistency is also small.

In the experiments, we used 100 randomly generated cases in the case-base, 200 randomly generated test-cases and repeated the setup 3000 times. In figure 5.9 the mean accuracies are plotted over k_1 and k_2. The baseline is the accuracy for the standard similarity measure which is about 96.7%.

Most importantly, the results suggest that the accuracy is close to 100% if at least one of the rules is correct. In this case, the other (inaccurate) rule does not have a big effect. We analyzed this situation in an additional experiment, where we kept $k_1 = 50$ and varied only k_2. The accuracy curve is depicted in figure 5.10. First of all, the accuracy is always above 99.8% accuracy. Moreover, if k_2 is far off the correct value, the accuracy is 100%. Only if k_2 is close to the correct value, the accuracy decreases slightly. On first sight, this finding appears counterintuitive, as the accuracy is best if there is great in-

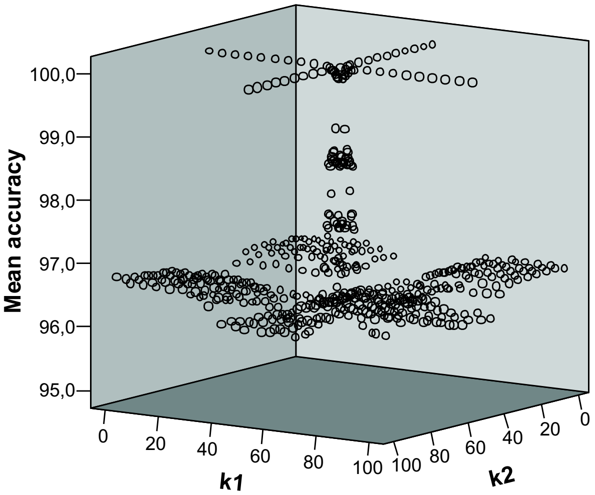

Figure 5.9: Accuracy of a similarity measure with two virtual attributes $A_v(c) \leftarrow A_1(c) > k_1$ and $A_w(c) \leftarrow A_1(c) > k_2$ for the concept $f(c) \leftarrow A_1(c) > 50$ in a two-dimensional universe. k_1 and k_2 are sampled at intervals of 5, except in the interval $[45, 55]$ where data is sampled in intervals of 1.

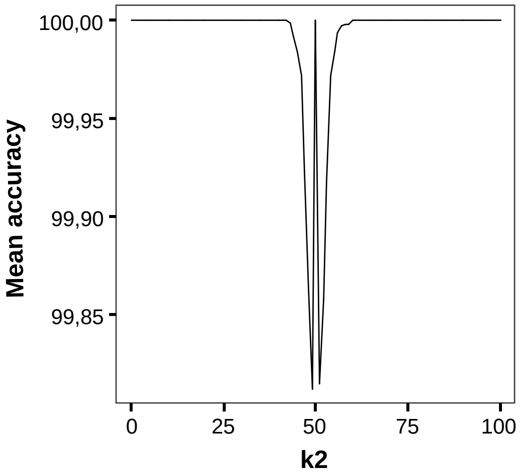

Figure 5.10: Accuracy of a similarity measure with a correct virtual attribute $A_v(c) \leftarrow A_1(c) > 50$ and an inaccurate virtual attribute $A_w(c) \leftarrow A_1(c) > k_2$ for the concept $f(c) \leftarrow A_1(c) > 50$ in a two-dimensional universe. Note that the accuracy of the standard similarity measure is much lower, at about 96.7.
]

consistency, and the accuracy is lowest if the inconsistency is rather small. So let us look at this finding in more detail. The fact that for very wrong values of k_2 the accuracy is almost 100% is closely related to the finding that very inaccurate virtual attributes do not effect classification accuracy negatively (see figure 5.3). Since the correct virtual attribute separates negative from positive cases, the very wrong virtual attribute is in a homogeneous region, that is, a region with cases that have equal classes. Thus, if k_2 is far off, to either side of it there are many equally classed cases, so that a misclassification is unlikely. Only if k_2 is close to the concept boundary and the correct virtual attribute, the situation can arise that there is no case in region Y between the inaccurate and the correct virtual attribute. If this happens, for a query q within Y there will be no case that has the same values as q for the virtual attributes. Thus, the area of equal similarity around q has to extend over at least one of the separating lines specified by the virtual attributes. This way, the negative and positive cases are not separated by the correct virtual attribute anymore, and a misclassification is possible. However, since the situation that Y is empty and that a query is located in Y is rare, the decrease in average accuracy in figure 5.10 is only small.

Back to the experiment where both k_1 and k_2 are varied: If we assume that both virtual attributes are inaccurate and k_1 is different from k_2, there can be two situations: Either both separating lines are on the same side of the concept boundary, or they are on different sides (see figure 5.11). With similar arguments as for inaccurate virtual attributes, we can treat queries in the different regions of each situation. If both virtual attributes are on the same side of the concept boundary, region A is a good region for queries (similar to region Z in figure 5.4). The area of equal similarity has to go over two separating lines before it reaches a case that has the wrong class. Also region B is a good region for queries. Either there is a case in region B, then it has the correct class, or there is no case in B. In the latter situation, the area of equal similarity will have to extend over one or even two separating lines before it reaches a case. Then there is the same probability for misclassifications as for the standard similarity measure. C is a bad region for queries, for the same reasons as region Y in figure 5.4. If region C is small and contains no case, there will be no misclassifications for queries in region D. However, if there are cases in C, misclassifications for queries in D are possible.

If the virtual attributes are on different sides of the concept boundary, region A and D are good regions for queries, as they are separated from the other classes by a separating line. Regions B and C are prone to misclassifications

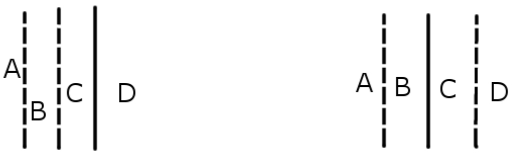

Figure 5.11: Situations for two inaccurate virtual attributes. Either the separating lines (dashed lines) described by the virtual attributes are on the same side of the concept boundary (solid line) or they are on different sides.

for the same reasons as region Y in figure 5.4.

To further understand the effect of inconsistent virtual attributes, we plot the accuracy of similarity measures with two virtual attributes, where one virtual attribute is held constant and the other is varied. In figure 5.12 the accuracy curves of three similarity measures are depicted. For the left curve the constant virtual attribute was $A_u(c) \leftarrow A_1(c) > 20$, for the middle curve it was $A_v(c) \leftarrow A_1(c) > 40$, and for the right one $A_w(c) \leftarrow A_1(c) > 48$. The constant values of 20, 40, 48 were chosen because they represent values for which the accuracy is similar to the standard similarity measure, worse than the standard measure, and better than the standard measure, respectively (refer to figure 5.3 to see the accuracy for measures with exactly one virtual attribute). An additional virtual attribute $A_x(c) \leftarrow A_1(c) > k_2$ was combined with each of A_u, A_v, A_w, and the accuracy of these three similarity measures is plotted over k_2.

The curves are similar to the accuracy curve for one virtual attribute (see figure 5.3). Note that the accuracy of using only the constant virtual attribute is equivalent to setting $k_2 = 0$ or $k_2 = 100$, which serves as baseline in this experiment. This way, we can analyze the effect of adding an inconsistent virtual attribute A_x to the extended similarity measure. Intuitively, adding an inconsistent virtual attribute to an already extended similarity measure is similar to adding an inaccurate virtual attribute to a standard similarity measure.

Obviously, the accuracy is increased for all measures if k_2 is close to the

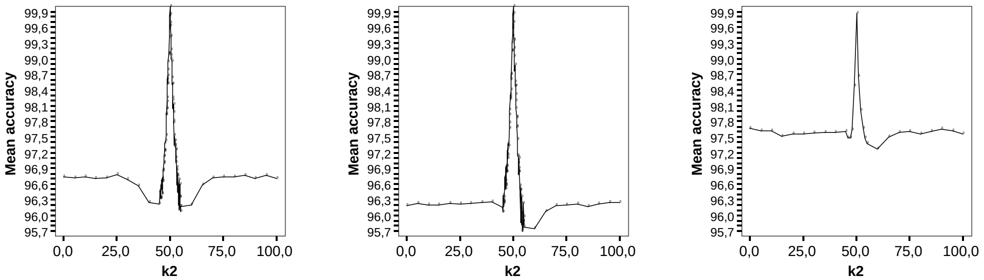

Figure 5.12: Accuracy for a similarity measure with two virtual attributes, where one attribute is held constant ($A(c) \leftarrow A_1(c) > k$ with $k = 20$ on the left, $k = 40$ in the middle, $k = 48$ on the right), and the other is varied as $A_x(c) \leftarrow A_1(c) > k_2$ with k_2 plotted on the horizontal axis.

concept boundary. As already noted, even accuracy rates close to 100% are possible if k_2 is close to the correct value. Yet, also decrease in accuracy is possible. To analyze the low accuracy situations, we have to distinguish between the situation where both virtual attributes are on the same side of the concept boundary and the situation where they are on different sides, because obviously the accuracy curves are not symmetric with respect to the position of the concept boundary. If both virtual attributes are on the same side of the concept boundary (that is, $k < 50$), accuracy degradation only occurs if the second virtual attribute is closer to the concept boundary than the constant one. Furthermore, the degradation can only occur if the second virtual attribute is close enough to the concept boundary (that is, it must be in the valley in figure 5.3).

If the virtual attributes are on different sides of the concept boundary, the same degradation valley as for single inaccurate virtual attributes occurs.

5.2.3 Partial and inaccurate knowledge

Partial knowledge can be visualized as gaps in the domain theory. In figure 5.13 a partial domain theory for a polygon-shaped concept is depicted. The nodes correspond to attributes, and the arcs denote relationships, that is, the more abstract attribute is derived from the more primitive attributes. Observables (here X and Y) are located at the bottom, the classification goal is at the top, and in between are intermediate attributes. A domain theory

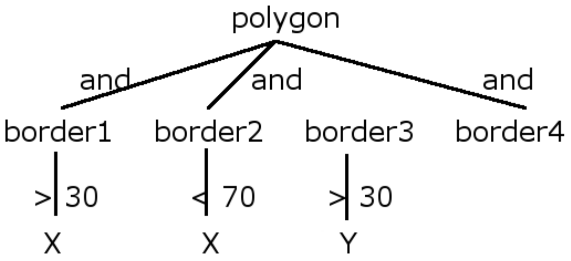

Figure 5.13: A partial domain theory for describing a polygon-shaped concept.

is partial if there are gaps. A gap in the bottom means that an intermediate attribute is not fully defined (in the example, *border*4 has a gap in the definition, because it is not connected to any observable). If there is a gap in the top, then the relation of an attribute to the classification goal is unknown (in the example, the relation of the fully defined *border*3 to the concept *polygon* is undefined). While gaps in the top of a domain theory can be bridged using weight learning methods, gaps in the bottom can not easily be bridged. In the following we use fully defined intermediate concepts as virtual attributes.

Intuitive analysis:

The effect of using a partial, but otherwise correct domain theory for deriving nominal virtual attributes can best be seen with a concept that has more than one boundary. In figure 5.14 (left) the misclassifications of a standard similarity measure for a centered concept $f(c) \leftarrow (30 < A_1(c) \wedge A_1(c) < 70) \wedge (30 < A_2(c) \wedge A_2(c) < 70)$ with the shape of a square in a two-dimensional universe are depicted. For a centered, square concept, the optimal weight setting is to weight both attributes equally (Ling & Wang, 1997). Thus, the misclassifications depicted are the best that can be achieved with a standard similarity measure. Unsurprisingly, the misclassifications occur around the concept boundaries.

If we add a partial chunk of knowledge (i. e. a part of the domain theory as a set of horn clauses) as the virtual attribute $A_v(c) \leftarrow 30 < A_1(c)$ to describe the left vertical concept boundary, the misclassifications at that boundary vanish. Furthermore, the probability of additional misclassifications at the

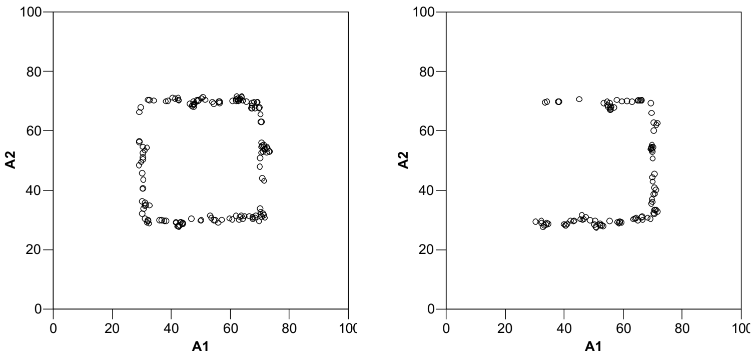

Figure 5.14: Misclassifications for the concept $f(c) \leftarrow (30 < A_1(c) \wedge A_1(c) < 70) \wedge (30 < A_2(c) \wedge A_2(c) < 70)$ with a standard similarity measure (left) and a similarity measure with the virtual attribute $A_v(c) \leftarrow 30 < A_1(c)$ (right).

line $A_1 = 30$ above and below the concept is small (see figure 5.14 (right)). This is a nice property as it allows to add partial knowledge chunks to approximate single target boundaries. Only at the corners where the horizontal target boundaries hit the vertical line, additional misclassifications are possible. In section 5.2.4 we will argue empirically that such additional misclassifications are so rare that they can be ignored (cf. (Ling & Wang, 1997) for a similar argument).

Thus, adding partial (but otherwise correct) knowledge as virtual attributes increases the classification accuracy monotonically. We have also shown this in (Steffens, 2004c). Figure 5.15 depicts the classification accuracy of a similarity measure with different numbers of partial, but correct virtual attributes. The target concepts were centered and square-shaped of the form $f(c) \leftarrow A_1(c) > LOW \wedge A_1(c) < HIGH \wedge A_2(c) > LOW \wedge A_2(c) < HIGH$ in a universe spanned by two numerical attributes A_1, A_2 in the range [0,100]. $(LOW, HIGH)$ was one of $\{(20, 80), (30, 70), (40, 60), (45, 55)\}$. 100 randomly generated cases were in the case-base, 200 random test cases were used, and 3000 runs were performed. In the experiments the sequence in which virtual attributes were added did not matter, so that only the number of virtual attributes is plotted (and not their identity, e. g. "left boundary and top boundary").

The results show that the effects of correct virtual attributes add up. For these target concepts the optimal weight ratio was used (i. e. all attributes were weighted equally), so that the standard similarity measure performs

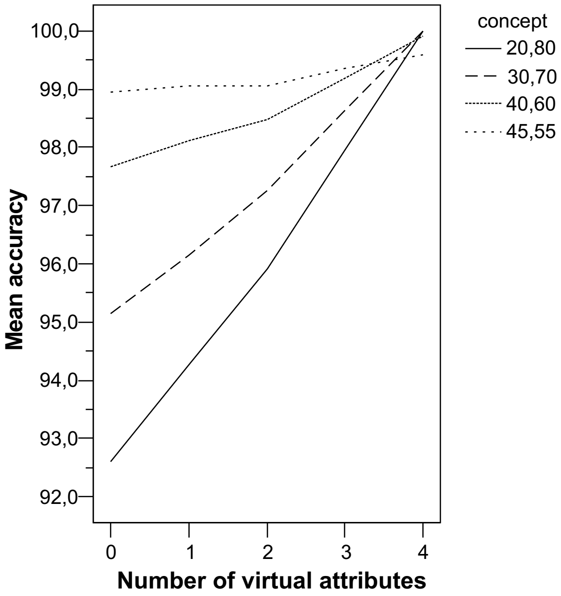

Figure 5.15: Accuracy of a similarity measure for different concepts with different numbers of concept boundaries described by correct virtual attributes.

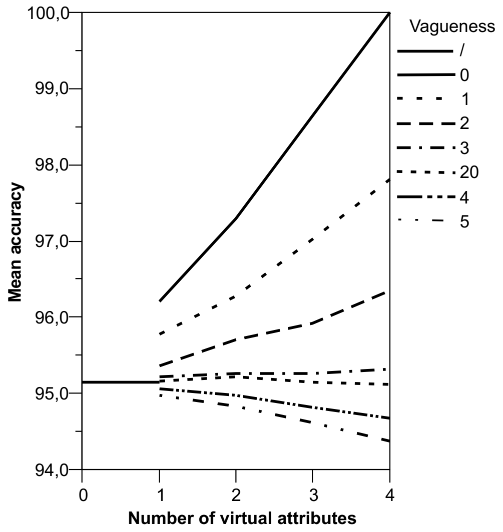

Figure 5.16: Accuracy of a similarity measure with different numbers of concept boundaries described by virtual attributes with different degrees (in percent of the correct value) of inaccuracy.

at its optimum. Still, with virtual attributes the accuracy was improved. Differences between the concepts are due to the boundary length. The smaller the concept area is, the smaller is the concept boundary and the smaller is the probability of errors.

However, this experiment assumed that the virtual attributes were correct. In order to test whether the effects of virtual attributes add up even if they are inaccurate, we ran another experiment. We used the target concept $f(c) \leftarrow A_1(c) > 30 \wedge A_1(c) < 70 \wedge A_2(c) > 30 \wedge A_2(c) < 70$, and varied the inaccuracy of the virtual attributes from 0 to 5% of the correct value. In figure 5.16 the accuracy curves are plotted.

If the attributes' inaccuracy is small (for example, 1 or 2%), the classification accuracy increases as more virtual attributes are added. However, if the attribute's inaccuracy is within a certain interval (i. e. 4 or 5%), the classification accuracy degrades as more virtual attributes are incorporated. We tested the classification accuracy curve for single inaccurate virtual attributes for this concept, and it turned out that the degradation occurs for inaccuracy levels that are in the classification accuracy valley of single inaccurate virtual attributes (see figure 5.3). Accordingly, if the attribute's inaccuracy level is rather high (e. g. 20% off) (that is, in the value range for which single inaccurate attributes perform like the baseline), the degradation does not occur anymore, but the classification accuracy remains rather constant. This means that both the positive and negative effects of virtual attributes add up in a straight-forward way.

Formal analysis:

In order to describe the effect of partial knowledge formally, we extend our calculation of the error rate from section 5.2.1. Remember that E_{total} describes the error introduced by a unit-length boundary. Thus, the error rate for a concept described by a n-sided polygon $l_1, l_2, \ldots, l_n$ can be calculated as

$$length(l_1) \cdot E_{total} + length(l_2) \cdot E_{total} + \ldots + length(l_n) \cdot E_{total}$$

(cf. (Ling & Wang, 1997)).
For a concept boundary that is not described by any virtual attribute, E_{total} can be computed in the way originally proposed by Ling and Wang:

$$E_{LW} = 2 \cdot \int_0^\infty \left(\int_0^\infty \left(1 - e^{-\lambda A_1}\right) \cdot e^{-\lambda A_2} dd_3 \right) dd_2$$

where

$$A_1 = ArcCos\left(\frac{d_2}{d_2 + d_3}\right)(d_3 + d_2)^2 - d_2 \cdot \sqrt{(d_3 + d_2)^2 - d_2^2}$$

$$A_2 = \pi \cdot (d_3 + d_2)^2 - A_1$$

To experimentally validate these formal predictions, we varied the number of virtual attributes and their inaccuracy. The target concept was $f(c) \leftarrow A_1(c) > 40 \wedge A_1(c) < 60 \wedge A_2(c) > 40 \wedge A_2(c) < 60$, the virtual attributes described the four concept boundaries. The number of virtual attributes in the similarity measure was varied from 0 to 4. Furthermore, the inaccuracy of the virtual attributes was varied: We ran experiments with correct virtual attributes (inaccuracy of 0%), with attributes that were 2% off, and with attributes that were 15% off. These values were chosen to reflect the situation of partial, but correct knowledge (0% inaccuracy), of knowledge that is only slightly inaccurate and increases classification accuracy (2% inaccuracy), and of knowledge that is so inaccurate that it decreases classification accuracy (15% inaccuracy). The classification accuracy curves for the formal prediction

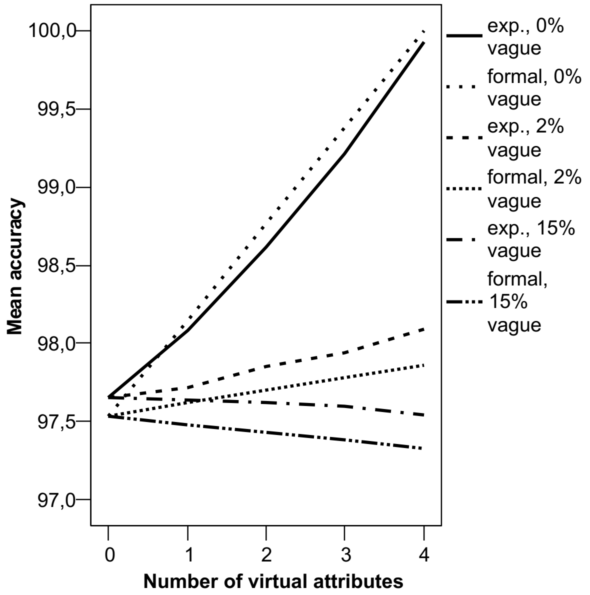

Figure 5.17: Experimental and formal accuracies for different numbers of virtual attributes with inaccuracy levels of 0, 2, and 15%. The target concept was in the shape of a centered square with side length 20. For the experimental data, 100 random cases were in the case-base, 200 random test-cases were used and 3000 runs were averaged.

and the experimental results are shown in figure 5.17. Apparently, for all three situations, the corresponding curves fit to a certain degree. The formal predictions are linear when plotted over the number of virtual attributes, and also the experimental data suggest a linear function.

5.2.4 Conjunctions of virtual attributes

Describing a concept boundary with a nominal virtual attributes leads to the fact that the boundary is approximated by a separating line that partitions the whole instance space, even if the concept boundary is only short. As we have mentioned in section 5.2.3, this can lead to misclassifications at corners where the other concept boundaries intersect the separating line. Even worse, such corners at intersections are multiplied if more than one concept boundary is approximated with separating lines. In this subsection we investigate empirically whether such misclassifications can be avoided by combining the virtual attributes into a conjuncted attribute if the target concept is convex.

Assume there is a set of binary virtual attributes $A_1, A_2, \ldots, A_n$ that are used to approximate n concept boundaries. If the target concept is convex, they can be combined into another virtual attribute $A(C) \leftarrow A_1 \wedge A_2 \wedge \ldots \wedge A_n$. If this combined virtual attribute is added to the standard similarity measure instead of the single A_i, the separating lines do not cross the whole instance space but are constrained by the other separating lines. The prediction is that the error rate is decreased (as compared to the measure with n virtual attributes), because the instance space is not separated superfluously by overlong separating lines.

Unfortunately, this prediction is not met by the experimental results. A counter-example is the following setup. The target concept has again the shape of a centered square with side length 40 in an instance-space spanned by two numerical attributes A_5, A_6 with the range [0,100]. One similarity measure sim_1 incorporates three correct virtual attributes $A_1(c) \leftarrow A_5(c) > 30$, $A_2(c) \leftarrow A_5(c) < 70$, $A_3(c) \leftarrow A_6(c) < 70$. Another similarity measure sim_2 incorporates only the combined virtual attribute $A(c) \leftarrow A_1(c) \wedge A_2(c) \wedge A_3(c)$. With the usual setup of 100 cases in the case-base, 200 test cases, 3000 runs, and equal weights the accuracies are 98.6177 (standard error 0.1804) for sim_1 and 98.6278 (standard error 0.1793) for sim_2. The difference (for N=3000) is not significant in a two-tailed t-test (p=0.689). We believe this is due to the fact that the additional misclassifications introduced by overlong separating lines are very rare. This is in line with arguments in (Ling & Wang, 1997) and is a promising result for the use of partial virtual attributes describing concept boundaries.

5.2.5 Experiments with real-world data

The domain of the previous sections allowed us to vary the inaccuracy and partialness of the domain theory. However, since the domain was handcrafted and simple, we ran additional experiments with real-world data from the UCI Machine Learning Repository (Blake & Merz, 1998). We used a data set that provides an imperfect domain theory. Note that some data sets in the repository come along with perfect domain models, as the instances were created by those models. However, in this thesis we use only data sets whose domain theories are imperfect.

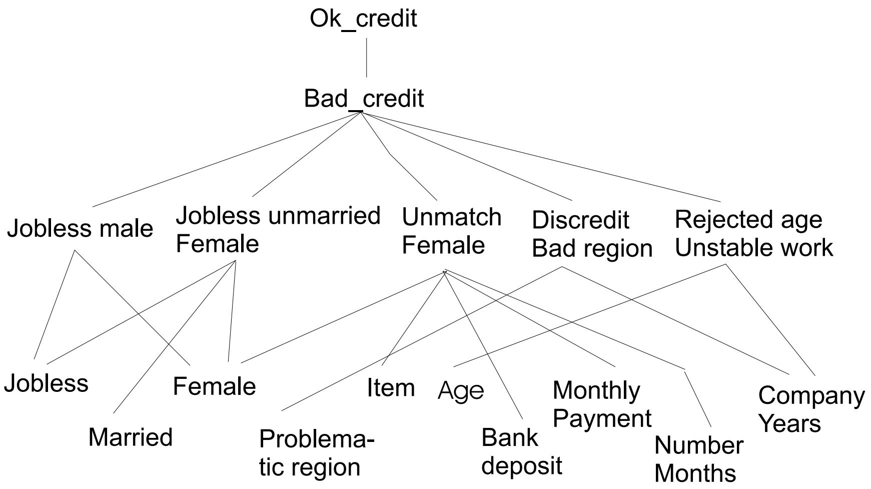

Figure 5.18: The structure of the domain theory of the JCS domain. Nodes denote attributes, arcs denote relations. The leaf nodes are observables, the top node is the classification goal. The attributes in between are intermediates.

We presented preliminary results already in (Steffens, 2004b) and (Steffens, 2005c) and will extend them here.

The Japanese Credit Screening (JCS) domain provides 124 cases of credit applications. There are 5 binary, 5 linear, and the binary target attribute (whether the credit was granted or not). The JCS domain comes with a domain theory that was created by interviewing domain experts. Since such knowledge acquisition is difficult, the theory is imperfect and classifies only 100 of the 124 cases correctly. The theory consists of PROLOG predicates specifying 7 binary intermediate concepts, ranging from simple conjuncted observables (e. g. `jobless_male`) to complex concepts using arithmetic (e. g. $\texttt{numberMonths} \cdot \texttt{monthlyPayment} > \texttt{bankDeposit}$). The structure of the theory is depicted in figure 5.18. All intermediate concepts are related to the goal and are completely defined, so that all intermediates were treated as candidates for virtual attributes.

Most of the intermediate concepts process several observables. For example, *rejected_age_unstable_work* processes the observables *age* and *number_years*[2]:

[2] This attribute denotes the number of years that the applicant worked at the same company.

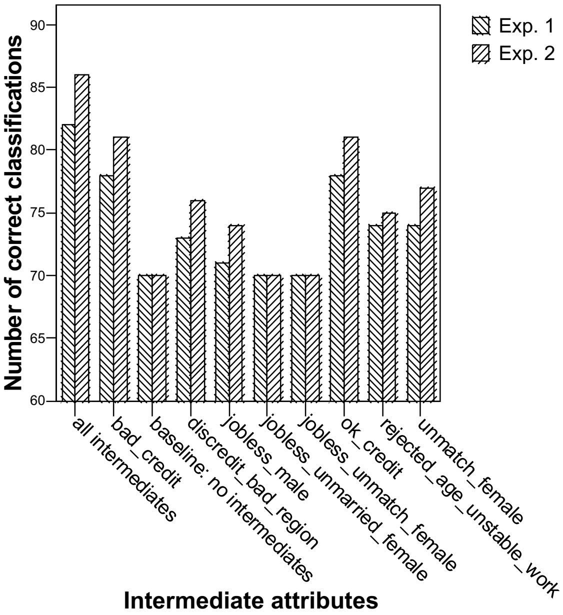

Figure 5.19: Number of correct classifications of similarity measures containing different virtual attributes. In experiment 1, virtual attributes were weighted with 1, in experiment 2 with 10. The case-base contained 124 cases.

```
rejected_age_unstable_work(S) :-
    age_test(S, N1),
    59 < N1,
    number_years_test(S, N2),
    N2 < 3.
```

To generate a similarity measure for the JCS domain, one or all intermediates were added as virtual attributes. In the first experiment all attributes were weighted equally. In the second round the virtual attributes were weighted ten times greater than the observables, because in the first round the virtual attributes were overshadowed by the large number of observables. Thus we chose the weight that makes the virtual attribute as heavy as the observables together. The classification accuracies of the resulting measures were achieved using the leave-one-out design (S. M. Weiss & Kulikowski, 1991). That is, each of the cases in the case-base is selected as query once and the others are used to predict its class. The accuracies are depicted in figure 5.19.

The results suggest that adding intermediate concepts as virtual attributes

can indeed improve the classification accuracy, when compared to the standard similarity measure that does not use intermediates. This supports our finding in the artificial domain that imperfect knowledge used as virtual attributes can increase classification accuracy. Unfortunately, the accuracies are worse than the accuracy of the domain theory alone. Furthermore, some intermediate concepts did not improve the baseline. This means that we have to find methods to select good intermediates from the domain theory.
In the next chapter we will show how to select intermediates so that the classification accuracy is even more increased.

5.3 Numerical matching knowledge

5.3.1 The virtual attribute method and the equivalence method

Numerical matching knowledge defines regions in the instance space in which cases are believed to be classified identically. These regions are defined by intervals on the attributes. Vagueness of numerical matching knowledge means that the positions of the onset and offset of the interval are not precisely known. If a value is selected, it is likely to be inaccurate.
Remember that numerical matching knowledge can be incorporated with the equivalence method and with the virtual attribute method. We compare the effects of these two incorporation methods by assuming inaccurate knowledge about the expanse of the concept on attribute A_1.
For the experiments, the concept is (for making weighting transparent) again a centered square, more precisely $f(c) \leftarrow A_1 > 30 \wedge A_1 < 70 \wedge A_2 > 30 \wedge A_2 < 70$. According to the equivalence method, the local similarity function for A_1 is

$$d_1(c_n, c_m) = \begin{cases} 1 & : \; iff\ A_1(c_n) \in R \wedge A_1(c_m) \in R \\ (1 - \frac{|A_1(c_n) - A_1(c_m)|}{range(A_1)})^2 & : \; else \end{cases}$$

where R is defined as $[50 - k, 50 + k], 0 \leq k \leq 50$ (to allow for variation of inaccuracy via k). That means, that the center of the estimated interval is in the center of the concept, and its range can be varied via k.
The virtual attribute method defines an attribute $A_3 \leftarrow A_1 > 50 - k \wedge A_1 < 50 + k$ and adds it to the standard similarity measure in the usual way.

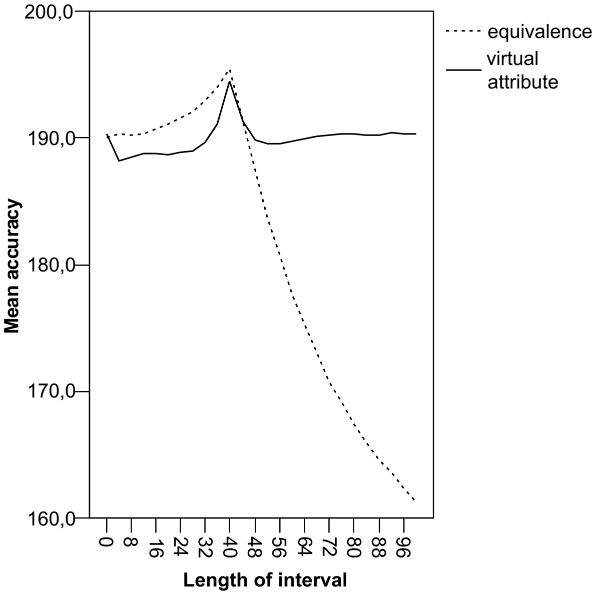

Figure 5.20: Comparison of the equivalence method and the virtual attribute method for numerical matching knowledge (accuracy is given as number of correctly classified cases, not in percent).

Figure 5.20 depicts the accuracies of the equivalence method and the virtual attribute method for the concept $f(c)$. Apparently, the two methods are not equivalent, although they use the same knowledge. But since the assumptions of the two incorporation methods are different, also their effect on classification is different. The equivalence method increases classification accuracy as long as the estimated interval is inside the concept (the baseline of the standard similarity measure is equivalent to using an interval length of 0). As soon as k is greater than 20 (i. e. the interval length is 40), the accuracy decreases rapidly. And for large intervals, the accuracy is dramatically worse than the baseline. Practically this means that when in doubt a CBR designer should tend to use smaller intervals for the equivalence method.

The behavior of the virtual attribute method is different. It does not increase monotonically for lengths smaller than 20 as the equivalence method does. That is, if the estimated interval is small, the accuracy is decreased. However, the maximum of its accuracy curve is the same as for the equivalence method. The virtual attribute method is more robust against very inaccurate knowledge in the form of large intervals. Its minimal accuracy is significantly better than the minimal accuracy of the equivalence method.

It is apparent that the equivalence method increases accuracy even if the estimated interval for A_1 is not even close to the concept boundaries for A_1. So how does this accuracy increase come about? Setting the local similarity of all values in interval $R \subset A_1$ to 1 is equivalent to infinitely stretching dimension A_1 in R. This again is equivalent to using a local weight of 0 for A_1 in interval R. Thus, the decision of whether a query is in the concept or not is based on the other attributes, in this case A_2. This means, that if R is completely inside the target concept, the accuracy is increased, because A_1 is irrelevant for queries in R and only the other attributes are used.

The above experiment is satisfying for estimating the effects of the length of the estimated interval. However, its setup may appear artificial, as it assumes that the estimated interval is centered at the center of the target concept. Thus, in the following experiments, the onset of the interval is varied, while its length is held constant.

Figure 5.21 shows the accuracy of the two incorporation methods for the same target concept as before. However, this time the length of the estimated interval is held constant, and the onset of the interval is varied. Both incorporation methods have peaks if the onset or the end of the interval touches a concept boundary. This is due to the fact that misclassifications on the side of the boundary where the interval is are improbable. Since cases

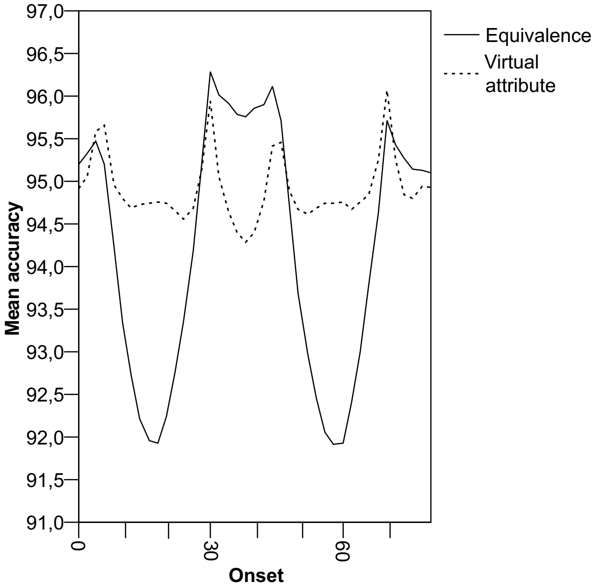

Figure 5.21: Accuracies of two similarity measures that incorporate an interval with length 25 at various onsets using the equivalence method and the virtual attribute method.

with the same class already have a local similarity of 1 for A_1 (in the equivalence method) or an additional local similarity of 1 if they are in the interval (in the virtual attribute method), it is very probable that a case is retrieved that has the same class as the query.

As in the previous experiment, the equivalence method yields good accuracies as long as the interval is completely within the concept. However, if the interval goes over a concept boundary, the accuracy drops dramatically. In contrast, for the virtual attribute method the difference between the minimum and maximum accuracy is small, but the peaks are restricted to be around the concept boundaries.

Both incorporation methods perform best if the interval boundaries touch the concept boundaries. This finding is in line with the previous experiment. The reason for this is that cases that are on opposite sides of the concept boundary will be less similar than cases that are on the same side of the concept boundary, because of the equivalence or the additional virtual attribute, respectively.

5.3.2 Matching knowledge for distributed concepts

Distributed concepts are those that have at least two unconnected regions in the instance space. Such concepts are due to disjunctions in their definition (Ragavan & Rendell, 1991). Disjunctive concepts are problematic because they contradict the assumption of similarity-based classification which is that neighboring regions tend to be classified identically (continuity assumption, e. g. (Matheus, 1991)). Furthermore, disjunctive concepts tend to have longer boundaries for the same concept volume. Since misclassifications occur at concept boundaries (Ling & Wang, 1997), for disjunctive concepts classification accuracy is low. In this section we show how virtual attributes and matching knowledge can be applied to distributed concepts.

In our experiments we found that the benefit of both incorporation methods depends on how much overlap there is between the projections of the concept parts onto the axes. An example for a concept that has two non-overlapping parts is $f_1(c) \leftarrow (20 \leq A_1(c) \leq 40 \wedge 20 \leq A_2(c) \leq 40) \vee (60 \leq A_1(c) \leq 80 \wedge 60 \leq A_2(c) \leq 80)$[3]. The projections on both dimensions A_1, A_2 do not overlap (see figure 5.22 (left)). With our standard setting of 100 cases in the

[3] For readability's sake we combine the comparison operators slightly different from the definition of our domain language, and do not use one literal per comparison operator.

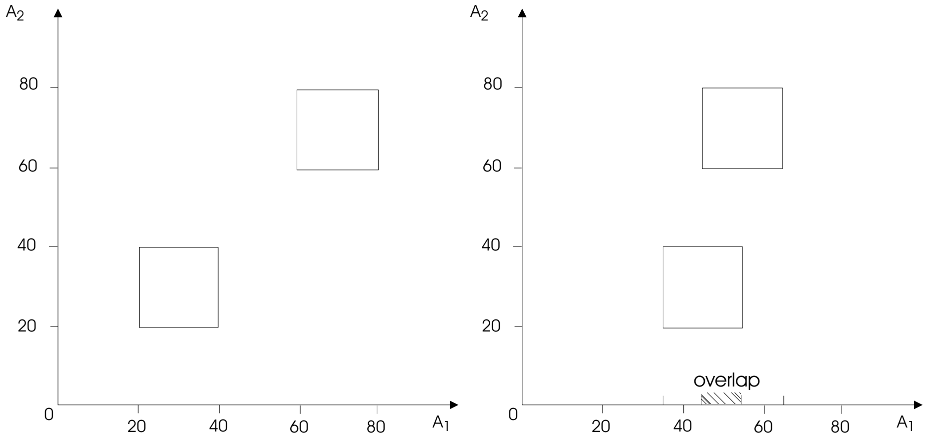

Figure 5.22: Concepts where the projections of the disjunctive parts onto the axes do not overlap (left) or overlap on A_1 (right).

Table 5.1: Accuracies of different similarity measures for a non-overlapping disjuncted concept.

Measure	Accuracy	Deviation
standard	95.40 %	1.75
with virtual attributes $A_{v,1}(c)$ and $A_{v,2}(c)$	96.94 %	1.65
with virtual attribute $A_w(c)$	96.14 %	1.68
with intervals $[20, 40]$ and $[60, 80]$ on A_1	96.69 %	1.70
with interval $[20, 80]$ on A_1	88.47 %	2.62

case-base, 200 test cases, 5000 runs and equal weights, different similarity measures yield the accuracies shown in table 5.1.
The virtual attributes are defined as follows:

$$A_{v,1}(c) \leftarrow 20 \leq A_1(c) \leq 40 \quad \mathit{and} \quad A_{v,2}(c) \leftarrow 60 \leq A_1(c) \leq 80$$

$$A_w(c) \leftarrow 20 \leq A_1(c) \leq 80$$

The intervals are incorporated using the equivalence method for numerical attributes. All measures are pairwise significantly different ($p < 0.001$ in a two-tailed t-test).
The measure using the virtual attributes $A_{v,1}, A_{v,2}$ yields a better accuracy than the standard measure, because misclassifications will only occur at

Table 5.2: Accuracies of different similarity measures for an overlapping disjuncted concept.

Measure	Accuracy	Deviation
standard	95.29 %	1.81
with virtual attributes $A_{u,1}(c)$ and $A_{u,2}(c)$	95.54 %	2.09
with virtual attribute $A_r(c)$	96.95 %	1.81
with intervals $[35, 55]$ and $[45, 65]$ on A_1	94.63 %	1.88
with interval $[35, 65]$ on A_1	93.16 %	2.03

the horizontal boundaries. However, since the vertical boundaries are rather short, the difference to the standard measure is not very large.

Surprisingly, even the measure using the inaccurate virtual attribute A_w is significantly better than the standard measure. They prevent misclassifications at the boundaries $A_1 = 20$ and $A_1 = 80$ and obviously do not introduce too many additional misclassifications. In contrast, if the equivalence method uses the interval from 20 to 80, the accuracy decreases drastically. This is due to the fact that the important attribute A_1 is basically made irrelevant in the interval 20 to 80, so that many misclassifications (in and outside the concepts) are introduced in that interval. However, if the interval is split into $[20, 40]$ and $[60, 80]$, the accuracy is almost as good as the accuracy of the measure with $A_{v,1}, A_{v,2}$.

Let us now consider a concept, where the projections of dispersed parts onto A_1 overlap. The test concept is $f_1(c) \leftarrow (35 \leq A_1(c) \leq 55 \wedge 20 \leq A_2(c) \leq 40) \vee (45 \leq A_1(c) \leq 65 \wedge 60 \leq A_2(c) \leq 80)$. Obviously, the two disjuncted parts overlap in the interval $A_1 = [45, 55]$ if projected onto A_1 (see figure 5.22 (right)). Again, we tested several similarity measures (see table 5.2).

The virtual attributes are defined as follows:

$$A_{u,1}(c) \leftarrow 35 \leq A_1(c) \leq 55 \quad \textit{and} \quad A_{u,2}(c) \leftarrow 45 \leq A_1(c) \leq 65$$

$$A_r(c) \leftarrow 35 \leq A_1(c) \leq 65$$

Most apparently, the measure using two virtual attributes, which performed best for the non-overlapping concept, now performs only slightly better than the standard measure. The difference is still significant (p=0.001). The decreased impact is due to the fact that both virtual attributes now separate even cases that belong to the same concept part, because the line that approximates one part's boundary goes through the other concept part. The measure that uses only one virtual attribute now performs better than the one with two virtual attributes, because all members of the concepts share the same value for A_r. Both similarity measures achieve higher accuracies than the standard measure, because they prevent misclassifications at the outer concept boundaries at $A_1 = 45$ and $A_1 = 65$.

As another difference to the non-overlapping concept, the measure with two intervals now performs worse than the standard measure. This is due to the fact that in both intervals there are two vertical concept boundaries ($A_1 = 55$ in $[45, 65]$ and $A_1 = 45$ in $[35, 55]$) which are ignored due to the equivalence statements.

Summarizing, both the virtual attribute and the equivalence method work better if the projections of the distributed concept parts onto the axes do not overlap.

5.4 Nominal Matching Knowledge

The incorporation method for nominal matching knowledge introduced in section 4.2.2 states the equivalence of two nominal attribute values. In section 5.4.1 we will investigate how this method performs if the attribute's domain has more than two values. In section 5.4.2 the attribute's domain has exactly two values. In section 5.4.3 we compare the virtual attribute method to the equivalence method. In all sections we investigate the effect of correct and of wrong knowledge both formally and empirically.

5.4.1 Equivalence of individual values

Empirical experiments

With our approach two attribute values can be treated as equivalent if the target concept is disjunctive regarding these attribute values. Consider a universe spanned by the attributes $A_1 = \{a, b, c, d\}, A_2 = \{w, x, y, z\}$ and the concept $f(c) \leftarrow (A_1(c) = a \vee A_1(c) = b) \wedge A_2(c) = w$. It is apparent, that there is no difference between a and b in this concept definition. Thus, our hypothesis is that adding the matching knowledge rule $a \equiv b$ into the similarity measure can increase accuracy.

In real world scenarios, the concept definition will certainly not be available in a perfect form. If a partial domain theory contains a disjunction of two values of an attribute, it is worth investigating whether stating the equivalence of

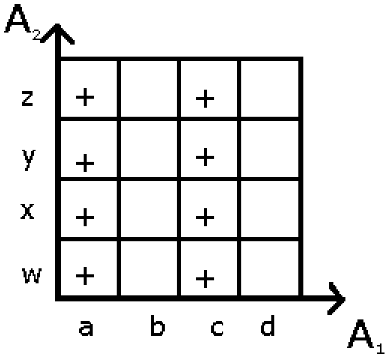

Figure 5.23: A concept in a two-dimensional universe, spanned by nominal attributes with a domain of four possible values. Squares that are filled with a + denote members of the concept. The attribute values a and c are equivalent, just as b and d, and all values of A_2.

the two disjuncted values is justified. Assume that a concept is conceptualized in the form

$$g(c) \leftarrow (A_v(c) \wedge A_2(c) = d) \vee A_w(c)$$

$$A_v(c) \leftarrow A_1(c) = a \vee A_1(c) = b$$

and $A_w(c)$ is undefined. Although keeping in mind that the knowledge is partial, there is no evidence that a and b should not be treated as equivalent. Yet, if additionally $A_w(c) \leftarrow A_1(c) = a \wedge \neg(A_2(c) = d)$ is known, this is a hint that the equivalence of a and b is not justified, since only a is used in the definition of A_w. However, from the knowledge engineer's point of view, it is also possible that the definition of A_w is incorrect, and should in reality be $A_w(c) \leftarrow (A_1(c) = a \vee A_1(c) = b) \wedge \neg(A_2(c) = d)$. Hence, when dealing with incomplete domain knowledge, there are many uncertainties. Analyzing the effect of incorrect equivalences in similarity measures is a first step to learn to handle such uncertainties.

Let us assume that for the target concept, the values $a, b \in A_i$ are indeed equivalent. A possible concept is depicted in figure 5.23.

With the standard un-weighted similarity measure for nominal attributes and a case-base I, a query q in an n-dimensional universe is classified correctly wrt. target attribute A_t, iff

- a) the query is in the case-base: $q \in I$, or
- b) the query is not in the case-base and all those cases in the case-base that differ in exactly one attribute from the query have the correct class: $\forall c : (\exists! i, \forall j : A_i(c) \neq A_i(q) \wedge (i \neq j \rightarrow A_j(c) = A_j(q))) \rightarrow A_t(c) = A_t(q)$, or
- c) the query is not in the case-base, there are no cases that differ in exactly one attribute from the query, and all cases in the case-base that differ in exactly two attributes from the query have the correct class: $\neg\exists c : (\exists! i, \forall j : A_i(c) \neq A_i(q) \wedge (i \neq j \rightarrow A_j(c) = A_j(q))) \rightarrow A_t(c) = A_t(q)$ and $\forall c : (\exists! h, \exists! i, \forall j : A_h(c) \neq A_h(q) \wedge A_i(c) \neq A_i(q) \wedge (h \neq i \wedge j \neq i \wedge j \neq h \rightarrow A_j(c) = A_j(q))) \rightarrow A_t(c) = A_t(q)$
- d) there are no cases that differ in exactly two attributes from the query, and all cases in the case-base that differ in exactly three attributes from the query have the correct class, and so on...

With an un-weighted similarity measure that states the equivalence of $a \in A_i$ and $b \in A_i$, a query is classified correctly, iff

- a) the query itself is in the case-base: $q \in I$, or
- b) the query is not in the case-base and there is a case in the case-base that differs from the query only in that one has value a for attribute A_i, and the other has value b and has the correct class: $\exists c \in I : A_i(c) \equiv A_i(q) \wedge \forall j : j \neq i \rightarrow A_j(c) = A_j(q)$ or
- c) all those cases in the case-base that differ in exactly one attribute from the query have the correct class: $\forall c : (\exists! i : (\neg(A_i(c) \equiv A_i(q)) \wedge \forall j : (i \neq j \rightarrow A_j(c) = A_j(q))) \rightarrow A_t(c) = A_t(q)$, or
- d) the above conditions do not hold and all cases in the case-base that differ in exactly two attributes from the query have the correct class, and so on...

That is, the effect of using an equivalence in a similarity measure is that the probability that an appropriate case is in the case-base is increased. For example, for maximal similarity it is not necessary that the query is included in the case-base. Rather, a case that differs from the query only in that it has an equivalent value for A_i will yield the maximal similarity. This

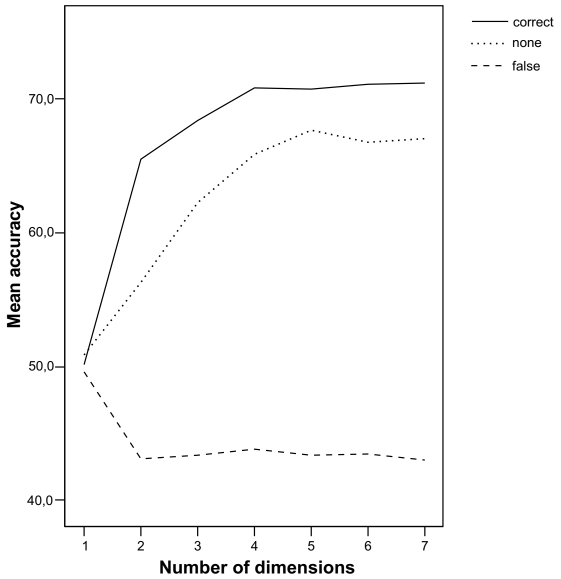

Figure 5.24: Accuracies of similarity measures that use correct, wrong, or no knowledge about equivalence.

finding is reminiscent to the result of (Ragavan & Rendell, 1991) for matching symmetrical tic-tac-toe board configurations. They stated the worth of the additional knowledge in terms of additional training instances.

In an experiment we analyze the difference between the standard similarity measure s, a measure s_c that uses a correct equivalence, and a measure s_w that uses a wrong equivalence. As the most basic set-up, we investigate the concept $f(c) \leftarrow A_1(c) = a_0 \vee A_1(c) = a_1$ in an n-dimensional universe spanned by the attributes $A_1 = \{a_0, a_1, a_2, a_3\}$ and $A_2, A_3, \ldots, A_n$ which all have the domain size 4. s_c uses the equivalence $a_0 \equiv a_1$, and s_w uses the equivalence $a_0 \equiv a_2$. We varied the number of dimensions and used a case-base that was filled to a quarter.The accuracies for the various conditions are depicted in figure 5.24.

Most apparently, using wrong knowledge about the equivalence of two attribute values is disastrous in this experiment. Although the accuracy remains rather stable when the number of dimensions is increased, it is always much lower than the accuracy of the standard similarity measure. Fortunately, using correct knowledge boosts the accuracy considerably.

In another experiment, we analyze the effect of nominal matching knowledge

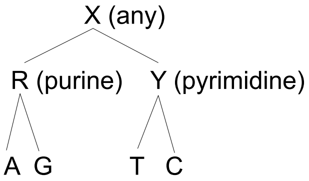

Figure 5.25: Hierarchy of nucleotides.

on a real-world data-set from the UCI Machine Learning Repository (Blake & Merz, 1998). In the "Promoter Gene Sequence" (PGS) data-set, there are 106 cases of promoter sequences, represented as strings of 57 nucleotides. For each case it is known whether it is a promoter gene or not. Nucleotides can be arranged in a hierarchy depicted in figure 5.25. The standard similarity measure will specify the similarity of identical nucleotides as 1, and of different nucleotides as 0. For the knowledge-rich similarity measure we use the equivalence $A \equiv G$ and $T \equiv C$. In a leave-one-out evaluation the standard similarity measure classifies 81 sequences correctly. Unfortunately, the knowledge-rich similarity measure classifies only 69 cases correctly. This degradation in classification accuracy is probably due to the fact that interpreting nucleotides that have a common father-node as equivalent is not appropriate for such a hierarchy. Although the nucleotides A and G might be more similar to each other than to T or C, stating their equivalence is too strong an assumption. Instead, the similarity between A and G should be stated as somewhere between 0 and 1.

Inspired from (Bergmann, 1998), we modified the integration of nominal matching knowledge into similarity measures, to allow for such a statement. The local similarity measure introduced in section 4.2.2 is modified as follows:

$$d_i(c_n, c_m) = \begin{cases} 1 & : \quad iff\ A_i(c_n) = A_i(c_m) \\ \gamma & : \quad iff\ A_i(c_n) \neq A_i(c_m) \wedge A_i(c_n) \equiv A_i(c_m) \\ 0 & : \quad else \end{cases}$$

For $\gamma = 0.5$ the classification accuracy increases to 87 cases. However, choosing a value for γ is similar to adjusting similarity values for nominal attributes as described in (Surma, 1994). Since this is a kind of distributional knowledge, fixing the value for γ is out of the scope of this thesis.

Formal analysis

In order to formally analyze the effect of adding an equivalence statement to a similarity measure, we adapt the average-case analysis for standard similarity measures by (Okamoto & Yugami, 2000) and extend it to similarity measures that use equivalence statements. The analysis assumes nominal attributes and a uniform distribution of the cases. The accuracy of a similarity measure for a classification f is calculated as a function of the following domain characteristics:

- n: the number of cases in the case-base (sampled independently of each other)
- r: the number of relevant attributes (used to define the concept)
- i: the number of irrelevant attributes
- l: the number of attribute values
- $\psi_f(d)$: Number of pairs of cases that have similarity d to each other and have the same class if the query does not have one of the equivalence values
- $\psi'_f(d)$: Number of pairs of cases that have similarity d to each other and have the same class if the query has one of the equivalence values

For sake of exposition, it is assumed that all attributes have the same domain size l. Any classification can be analyzed (e. g. disjunctive, conjunctive, or threshold classifications) by characterizing it by $\psi_f(d)$ and $\psi'_f(d)$. For better readability, we will not use all domain characteristics as parameters of the accuracy function, but implicitly assume them.

Our analysis is for a similarity measure that states the equivalence of two values $l_a, l_b \in A_e$ for an attribute A_e. In our analysis it is not necessary that this equivalence of the two values is reflected in the classification. In other words, our analysis handles both situations of correct and wrong equivalence statements. Let the universe $U(\mathbb{A})$ be spanned by the $m = r + i$ attributes. Let $S \subset U(\mathbb{A})$ denote the set of cases $\{c \mid A_e(c) = l_a \vee A_e(c) = l_b\}$, that is, the set of cases that have one of the values used in the equivalence statement. The target classification f is characterized by $\psi_f(d)$ and $\psi'_f(d)$, the number of pairs of cases that belong to the same class and have distance d to each other:

$$\psi'_f(d) = |\{\langle c_1, c_2\rangle \mid f(c_1) = f(c_2) \wedge s(c_1, c_2) = d \wedge c_1 \in S\}|$$

$$\psi_f(d) = |\{\langle c_1, c_2\rangle \mid f(c_1) = f(c_2) \wedge s(c_1, c_2) = d \wedge c_1 \notin S\}|\,,$$

where $c_1, c_2 \in U(\mathbb{A})$. Let $\Psi_f(c)$ be the set of cases that belong to the same class as c:

$$\Psi_f(c) = \{c' \in U(\mathbb{A}) \mid f(c) = f(c')\}.$$

According to our assumptions, each case is drawn with the same probability from the universe. We represent the distribution of cases as

$$p(c) = \frac{1}{|U(\mathbb{A})|} = \frac{1}{l^m}.$$

A query q is classified correctly, if a case t is retrieved that has the same class as the query. Thus, the classification accuracy for any f can be expressed as:

$$A(n) = \sum_{q \in U(\mathbb{A})} p(q) \cdot \sum_{t \in \Psi_f(q)} P_{nn}(t|n, q),$$

where $P_{nn}(t|n, q)$ is the probability that case t is selected as nearest neighbor for query q if the case-base contains n cases. Let d ($0 \leq d \leq m$) be the distance from an arbitrary query to its nearest neighbors. Then, $P_{nn}(t|n, q)$ can be split into two components:

$$P_{nn}(t|n, q) = P''^{d}_{nn}(d|n, q) \cdot P^{q}_{nn}(t|d),$$

where $P''^{d}_{nn}(d|n, q)$ is the probability that the nearest neighbor has similarity d to an arbitrary query q, and $P^{q}_{nn}(t|d)$ is the probability that among those nearest neighbors case t is selected (we assume here that ties are broken randomly).

Now, according to our classification algorithm if $s(q, t) \neq d$ then $P^{q}_{nn}(t|d) = 0$. Thus, we can rewrite the accuracy function as

$$\begin{aligned} A(n) &= \sum_{q \in U(\mathbb{A})} p(q) \cdot \sum_{t \in \Psi_f(q)} P''^{d}_{nn}(d|n, q) \cdot P^{q}_{nn}(t|d) \\ &= \frac{1}{l^m} \sum_{q \in U(\mathbb{A})} \sum_{d=0}^{m} P''^{d}_{nn}(d|n, q) \cdot \sum_{t \in \phi_f(q,d)} P^{q}_{nn}(t|d), \end{aligned}$$

where

$$\phi_f(q,d) = \{t \in \Psi_f(q) \mid s(q,t) = d\}\,.$$

The next step is to express $P''^{d}_{nn}(d|n,q)$ as a function of the domain characteristics. The number of cases with similarity d to query q depends on whether q has one of the equivalence values or not. We could use q as parameter to these functions, but since we want to express the accuracy function in a global way independent of individual queries, we define two functions, one for $q \in S$ and one for $q \notin S$. In the following, functions that assume $q \in S$ will have an apostrophe $'$. If $q \notin S$, the number of cases with similarity d to query q is

$$N_{dis}(d) = \binom{m}{d}(l-1)^{m-d}$$

If $q \in S$, the number of cases with similarity d to query q is

$$N'_{dis}(d) = \binom{m-1}{d}(l-1)^{m-d-1} \cdot (l-2) + \binom{m-1}{d-1}(l-1)^{m-d} \cdot 2$$

In other words, if the query does not have one of the equivalence values, there are $\binom{m}{d}$ ways to choose attributes in which the query and the case are identical. Multiplied with the ways in which the case can differ on the $m-d$ other dimensions (by having for each attribute one of the $l-1$ values that are different from the query's value), it yields the number of cases that have similarity d to the query q. If the query has one of the equivalence values, the similarity is more complicated. The first summand handles the situation that the case has identical values for other attributes than A_e. That is, d attributes can be chosen from the remaining $m-1$ attributes. Accordingly, the combinations in which the case differs has to be adapted: On the attributes that are neither A_e nor included in the d attributes, there are $l-1$ possibilities to differ. For attribute A_e there are $l-2$ possibilities to differ, since the two values l_a and l_b have to be excluded. The second summand handles the situation that case and query are identical in $d-1$ attributes and identical or equivalent in A_e. Accordingly, there are $d-1$ ways to select the attributes from the $m-1$ attributes excluding A_e. The factor 2 expresses that there are two ways to have similarity 1 for attribute A_e, by identity and by equivalence.

From our assumption of uniform distribution, it follows that the probability that an arbitrary case has similarity d from q is

$$P_{dis}(d) = \frac{1}{l^m} \cdot N_{dis}(d)$$

if $q \notin S$, and else

$$P'_{dis}(d) = \frac{1}{l^m} \cdot N'_{dis}(d).$$

Summing over d, the probability that an arbitrary case has a similarity smaller than d from q is:

$$P_{out}(d) = \sum_{e=0}^{d-1} P_{dis}(e)$$

if $q \notin S$, and else

$$P'_{out}(d) = \sum_{e=0}^{d-1} P'_{dis}(e).$$

Let us consider the situation that there are x nearest neighbors for q with similarity d and $n - x$ cases with a smaller similarity. The probability of this situation is

$$P^n_{nn}(d, x|q, n) = \binom{n}{x} \cdot P_{dis}(d)^x \cdot P_{out}(d)^{n-x},$$

if $q \notin S$, or

$$P'^n_{nn}(d, x|q, n) = \binom{n}{x} \cdot P'_{dis}(d)^x \cdot P'_{out}(d)^{n-x}$$

if $q \in S$.

Hence, the aforementioned probability that the nearest neighbor has similarity d to an arbitrary query q is

$$P^d_{nn}(d|n, q) = \sum_{x=1}^{n} P^n_{nn}(d, x|q, n) = \sum_{x=1}^{n} \binom{n}{x} P_{dis}(d)^x \cdot P_{out}(d)^{n-x}$$

if $q \notin S$, and

$$P'^d_{nn}(d|n, q) = \sum_{x=1}^{n} P'^n_{nn}(d, x|q, n) = \sum_{x=1}^{n} \binom{n}{x} P'_{dis}(d)^x \cdot P'_{out}(d)^{n-x}$$

if $q \in S$.

Now, we rewrite the accuracy function by splitting the sum over all cases into two sums, one sum for the cases in S and one for the other cases:

$$A(n) = \sum_{q \notin S} p(q) \sum_{d=0}^{m} P^d_{nn}(d|n, q) \sum_{t \in \Psi_f(q,d)} P^q_{nn}(t|d) +$$

$$\sum_{q\in S} p(q) \sum_{d=0}^{m} P'^{d}_{nn}(d|n,q) \sum_{t\in\Psi_f(q,d)} P'^{q}_{nn}(t|d)$$

$$= \sum_{d=0}^{m} \left(P^{d}_{nn}(d|n,q) \cdot \frac{1}{l^m} \cdot \sum_{q\notin S} \sum_{t\in\Psi_f(q,d)} P^{q}_{nn}(t|d) \right) +$$

$$\sum_{d=0}^{m} \left(P'^{d}_{nn}(d|n,q) \cdot \frac{1}{l^m} \cdot \sum_{q\in S} \sum_{t\in\Psi_f(q,d)} P'^{q}_{nn}(t|d) \right).$$

In this equation, let us define

$$P_f(d) = \frac{1}{l^m} \sum_{q\notin S} \sum_{t\in\Psi_f(q,d)} P^{q}_{nn}(t|d)$$

and

$$P'_f(d) = \frac{1}{l^m} \sum_{q\in S} \sum_{t\in\Psi_f(q,d)} P'^{q}_{nn}(t|d)$$

which clearly denotes the probability that a pair of cases belongs to the same class if they have similarity d to each other.
These probabilities can also be expressed as

$$P_f(d) = \frac{1}{l^m N_{dis}(d)} \psi_f(d)$$

and

$$P'_f(d) = \frac{1}{l^m N'_{dis}(d)} \psi'_f(d).$$

That is, the accuracy function can be expressed as

$$A(n) = h_1 \cdot \sum_{d=0}^{m} P^{d}_{nn}(d|n,q) \cdot P_f(d) + h_2 \cdot \sum_{d=0}^{m} P'^{d}_{nn}(d|n,q) \cdot P'_f(d),$$

where $h_1 = \frac{1}{l^m} \cdot (l-2) \cdot l^{m-1}$ and $h_2 = \frac{1}{l^m} \cdot 2 \cdot l^{m-1}$ are the number of cases that do not or do have one of the equivalence values, respectively.
Future work will have to analyze the effects of multiple equivalence statements. In section 5.4.2 we analyze multiple equivalence statements for binary attributes.

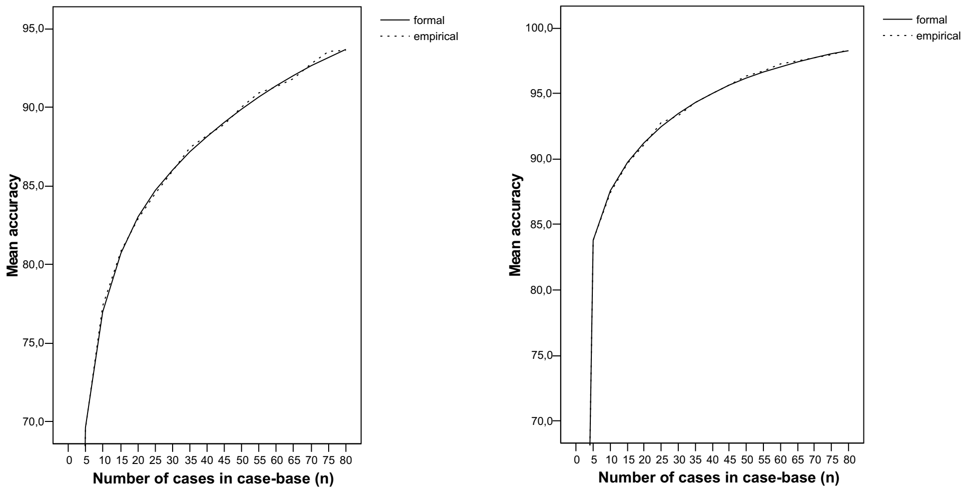

Figure 5.26: Comparison of accuracies obtained by experiments and by the formal analysis. Left: target concept is $f_1(c) \leftarrow A_0(c) = a_1 \vee A_0(c) = a_2$. Right: target concept is $f_2(c) \leftarrow (A_0(c) = a_1 \vee A_0(c) = a_2) \wedge A_1(c) = b_1$.

Empirical validation of the formal analysis

In order to test our assumptions and the analysis, we compare the accuracies obtained by experiments and by the formulas. First, we test two concepts in a universe spanned by the nominal attributes A_0, A_1, A_2, A_3, where each attribute has four possible values. The first concept is $f_1(c) \leftarrow A_0(c) = a_1 \vee A_0(c) = a_2$ and the second is slightly more complex: $f_2(c) \leftarrow (A_0(c) = a_1 \vee A_0(c) = a_2) \wedge A_1(c) = b_1$. The same similarity measure is used for both concepts and uses the correct equivalence statement $a_1 \equiv a_2$ for the local similarity of attribute A_0.

As depicted in figure 5.26, the formal results match the empirical ones so closely that the lines are hard to discern. As a general result, the accuracy increases monotonically if the number of cases in the case-base is increased. Furthermore, the second (more complex) concept is approximated slightly better. We believe that this is due to the fact that in the second concept the entropy is smaller.

In our next experiment, the similarity measure uses a wrong equivalence statement. We use the concept $f_1(c)$ again, but the similarity measure now states that $a_1 \equiv a_3$. Note that the only parameters that are different from the analysis with correct knowledge are the $\psi'_f(d)$ values. Figure 5.27 shows

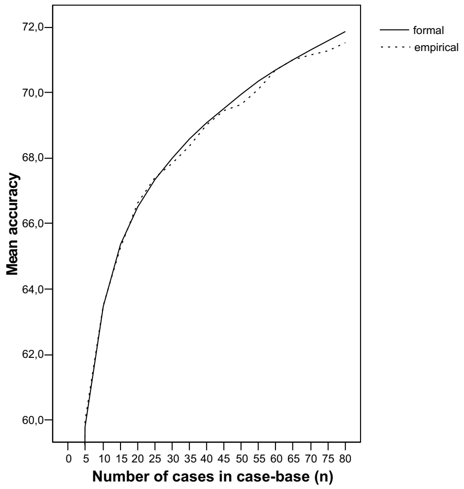

Figure 5.27: Comparison of accuracies obtained by experiments and by the formal analysis if the similarity measure uses a wrong equivalence statement.

the accuracies of the formal analysis and the empirical evaluation. Again, the curves match closely.

5.4.2 Stating irrelevance of an attribute

A special application of matching knowledge is to code the irrelevance of an attribute A_i. For numerical attributes this can be achieved by defining the whole domain of the attribute as the interval of equivalence, that is, $R = A_i$. For nominal attributes, this can be achieved by stating the equivalence of all possible value pairs of that attribute, that is, $\forall a_m, a_n \in A_i : a_m \equiv a_n$.

Correct matching knowledge:

Stating the irrelevance of an attribute via matching knowledge yields equivalent results to setting the attribute's weight to 0. The impact of filtering irrelevant attributes is already well-researched (e. g. (Aha, 1992; Wettschereck et al., 1997)). However, for completeness' sake we conducted some experiments to investigate the performance of using nominal matching knowledge to state the irrelevance of attributes. In these experiments, we used 50% of all possible cases in the case-base. The universe was spanned by $n = r + i$ binary attributes, of which $r \in \{1, 2, 3, 4, 5\}$ were relevant (that is, they were used

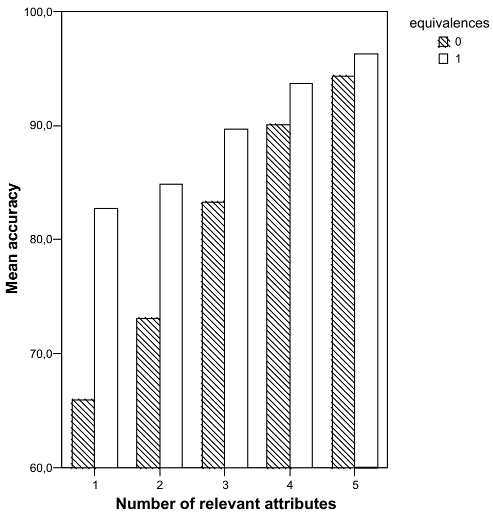

Figure 5.28: Accuracies of the standard similarity measure and of stating the irrelevance of an attribute via equivalence in a universe spanned by binary attributes. One attribute is irrelevant, the case-base contains 50% of all possible cases.

to define a random conjunctive concept) and $i \in \{1, 2, 3\}$ were irrelevant. In each experiment, s of the i attributes were stated as irrelevant using nominal matching knowledge. For each combination of i, r, s 3000 runs were executed with randomly generated concepts and randomly generated case-bases and test-instances. The mean accuracy for $i = 1$, $i = 2$ and $i = 3$ are shown in figure 5.28, 5.29 and 5.30, respectively.

These results are in line with previous research. The more attributes are known to be irrelevant, the better the accuracy. Furthermore, the effect of stating irrelevance of a given number of attributes decreases with the number of relevant attributes (that is, with the size of the universe).

To model these effects formally, we again extend the average-case analysis of (Okamoto & Yugami, 2000). By constraining the attributes to be binary, we can analyze the effect of several equivalence statements (not just one as in the previous section). The parameters of the accuracy function are simpler than in section 5.4.1: we only need the number of training cases n, the number of relevant (r) and irrelevant (i) attributes, and the number of pairs of cases that have similarity d to each other and are in the same class ($\psi_f(d)$). We assume that s equivalence statements are incorporated into the

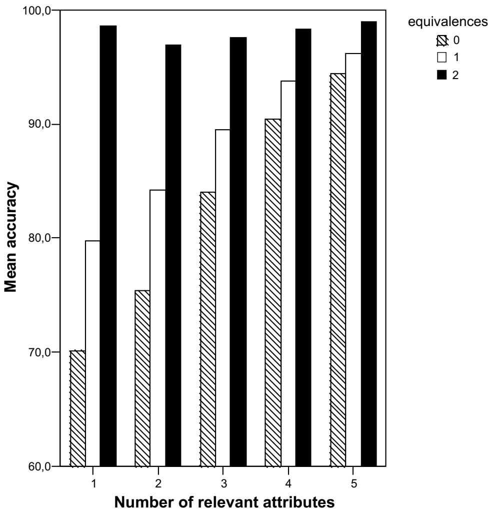

Figure 5.29: Accuracies of stating the irrelevance of various numbers (equivalences) of attributes via equivalence in a universe spanned by binary attributes. Two attributes are irrelevant, the case-base contains 50% of all possible cases.

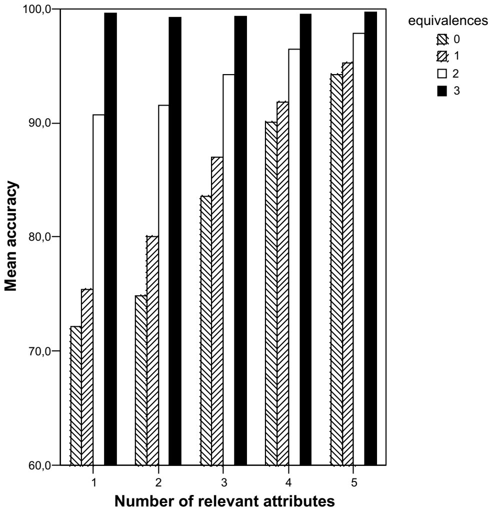

Figure 5.30: The same experiment, but three attributes are irrelevant.

similarity measure in order to state the irrelevance of s attributes. Due to the characterization of the concept by $\psi_f(d)$, the analysis holds for correct and for wrong equivalence statements.
With analogous steps as previously, we arrive at

$$A(n) = \frac{1}{2^m} \sum_{q \in U(\mathbb{A})} \sum_{d=0}^{m} P_{nn}^{d}(d|n,q) \cdot \sum_{t \in \phi_f(q,d)} P_{nn}^{q}(t|d),$$

where

$$\phi_f(q,d) = \{t \in \Psi_f(q) \mid s(q,t) = d\}.$$

The number of instances with similarity d from an arbitrary query is independent of q:

$$N_{dis}(d) = \binom{m-s}{d-s} \cdot 2^s$$

All pairs of cases will result in maximal local similarities for at least s attributes, thus $N_{dis}(d) = 0$ if $d < s$. For similarities d that are greater than or equal to s, there are $\binom{m-s}{d-s}$ many ways to select $d-s$ additional attributes from the $m-s$ remaining ones. Since the cases can have any of the two possible values for each of the s attributes, the factor 2^s is applied.
$P_{dis}(d)$ and $P_{out}(d)$ are analogous to the functions in section 5.4.1:

$$P_{dis}(d) = \frac{1}{2^m} N_{dis}(d)$$

$$P_{out}(d) = \sum_{e=s}^{d-1} P_{dis}(e)$$

Also the next functions are analogous:

$$P_{nn}^{n}(d,x|q,n) = \binom{n}{x} P_{dis}(d)^x P_{out}(d)^{n-x}$$

$$P_{nn}^{d}(d|n,q) = \sum_{x=1}^{n} \binom{n}{x} P_{dis}(d)^x P_{out}(d)^{n-x}$$

Thus, we finally arrive at

$$A(n) = \sum_{d=0}^{m} P_{nn}^{d}(d|n,q) \cdot P_f(d)$$

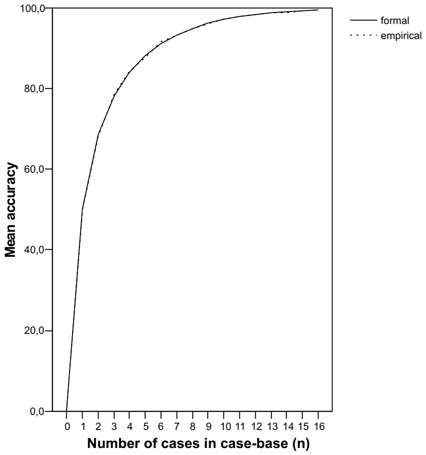

Figure 5.31: Accuracies of the empirical and formal analysis for the concept $f(c) \leftarrow A_0(c) = 0$ in a 4-dimensional universe, using equivalence statements for A_1 and A_2.

with $P_f(d)$ defined as before:

$$P_f(d) = \frac{1}{2^m N_{dis}(d)} \psi_f(d)$$

To validate our analysis empirically, we compare the formal predictions with experimental accuracies. In the first experiment, the target concept is $f_1(c) \leftarrow A_0(c) = "true"$ in a 4-dimensional universe. A_1 and A_2 are stated to be irrelevant by using two equivalence statements in the similarity measure. In the second experiment, the target concept is $f_2(c) \leftarrow A_0(c) = "true" \wedge A_1(c) = "true"$ in a 5-dimensional universe. A_2 is stated to be irrelevant by using an equivalence statement in the similarity measure. The formal and empirical accuracies are shown in figure 5.31 for experiment 1 and figure 5.32 for experiment 2. Apparently, the accuracies obtained in the empirical implementation are very similar to the formally predicted accuracies.

In the next subsection we will show that the formal and empirical accuracies match also for incorrect equivalence statements.

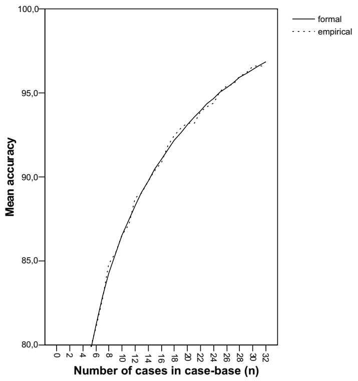

Figure 5.32: Accuracies of the empirical and formal analysis for the concept $f(c) \leftarrow A_0(c) = "true" \wedge A_1(c) = "true"$ in a 5-dimensional universe, using an equivalence statement for A_2. The scale is widened in order to show the small discrepancies due to noise in the empirical results. Note that the curve has in principle "reached" its asymptotically limit at the right hand-side, but due to the scaling of the vertical axis it appears as still increasing.

Incorrect matching knowledge:

We also investigate what happens if a relevant attribute is stated to be irrelevant via matching knowledge. The literature has only reported the effect of removing irrelevant attributes, but there is not much work (if any) on the effects of removing relevant attributes. Existing average-case analysis such as (Langley & Iba, 1993; Okamoto & Yugami, 2003, 2000) do not cover situations in which relevant attributes are believed to be irrelevant due to inaccurate domain knowledge. Our analysis presented in section 5.4.1 can be used to analyze the effect of one wrong equivalence statement, but not for several. Now, this analysis which is limited to binary attributes allows to model the effect of several wrong equivalence statements.

To compare the formal and empirical results, we used the concept $f(c) \leftarrow A_0(c) = "true" \wedge A_1(c) = "true"$ in a 4-dimensional universe and used two equivalence statements for A_1 and A_2 in the similarity measure. Apparently, the first equivalence statement is wrong. The empirically acquired accuracies depicted match the formal predictions (see figure 5.33).

In the next experiment we examine the effect of wrong nominal matching-knowledge for various situations. We used a universe spanned by 6 binary attributes. The target concepts are conjunctive and defined by r attributes of

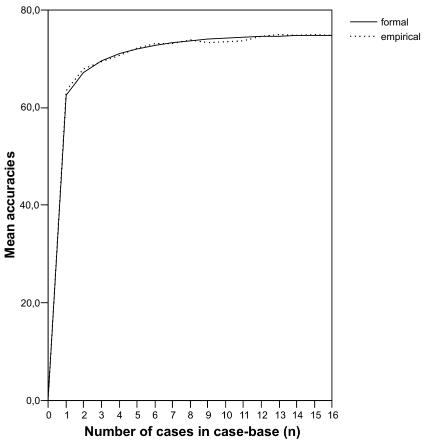

Figure 5.33: Accuracies of the empirical and formal analysis for the concept $f(c) \leftarrow A_0(c) = "true" \wedge A_1(c) = "true$ in a 4-dimensional universe, using a wrong equivalence statement for A_1 and a correct equivalence statement for A_2.

which s are practically removed by stating their values' equivalence to each other.

The results depicted in figure 5.34 are analogous to the experiment with correct matching knowledge. The only difference is that the accuracies are decreasing instead of increasing with increasing dimensions.

5.4.3 Comparing the equivalence method to the virtual attribute method

Experimental analysis:

As discussed in section 4.2.2, nominal matching knowledge can also be incorporated as virtual attribute. In this section the equivalence method as described in the previous experiments is compared to the virtual attribute method.

We analyze again the concept $f(c) \leftarrow A_1(c) = a_0 \vee A_1(c) = a_1$ in an n-dimensional universe where the domain size of all attributes is 4. We define a correct virtual attributes $A_c(c) \leftarrow A_1(c) = a_0 \vee A_1(c) = a_1$ and an inaccurate virtual attribute $A_w(c) \leftarrow A_1(c) = a_0 \vee A_1(c) = a_2$. We varied n and used a case-base that was filled to a quarter. The accuracies for the various

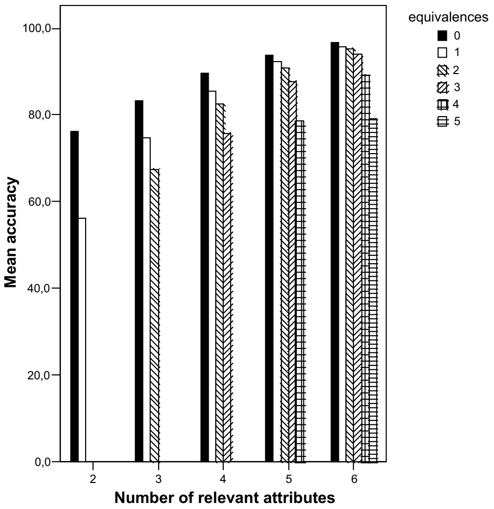

Figure 5.34: Accuracies of wrongly stating the irrelevance of different numbers of relevant attributes via equivalence in a universe spanned by 6 binary attributes. The case-base contains 50% of all possible cases.

conditions are depicted in figure 5.35.

Note that the curve for the standard similarity measure is in principle (apart from some noise) the same as in figure 5.24 where it is compared to measures using correct or wrong equivalence statements. Apparently, the measure with the virtual attribute A_c yields much higher accuracies than the standard measure or the equivalence method (refer to figure 5.24). Furthermore, even the similarity measure that uses the "wrong" virtual attribute A_w performs better than the standard measure. This is in stark contrast to the disastrous performance of the similarity measure with the wrong equivalence statement. So how does this come about?

First of all, the equivalence method has an effect only if the query and case in comparison both have one of the attribute values that are stated as equivalent. In contrast, virtual attribute are used for any query-case pair. This means that the similarity is increased if case and query both satisfy or both do not satisfy the virtual attribute. Thus, A_c makes also cases appear more similar that are both not members of the concept, whereas the equivalence method only makes concept members more similar to each other. This explains why the measure with the "wrong" virtual attribute A_w is better than the standard measure. Although A_w wrongly makes cases more similar if one

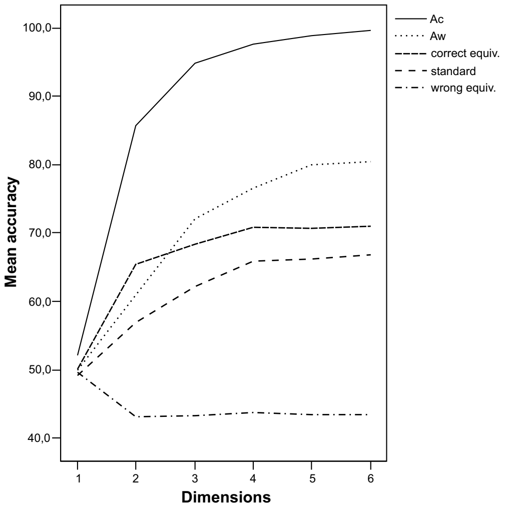

Figure 5.35: Accuracies of the standard similarity measure, the measure with the virtual attribute A_c and with A_w. For comparison, also the curves from figure 5.24 are depicted.

has a_0 and the other has a_2, it correctly makes cases more similar if one has attribute values a_1 and the other a_3. Now one might wonder why these effects of A_w do not cancel each other? The overall advantageous effect is that due to A_w also those cases become even more similar that are identical in the relevant attribute A_1 and only differ in A_2.

Formal analysis:

The virtual attribute method can also be formally analyzed for a given virtual attribute by extending (Okamoto & Yugami, 2000). The analysis is parallel to the one presented in section 5.4.1 with the following changes: Let the virtual attribute be $A_v(c) \leftarrow A_e(c) = a_{e1} \vee A_e(c) = a_{e2}$. The number of cases with similarity d to a query that does not have one of the values a_{e1}, a_{e2} for A_e is

$$N_{dis}(d) = \binom{m-1}{d}(l-1)^{m-1-d} \cdot 2+$$

$$\binom{m-1}{d-2}(l-1)^{m-d+1} + \binom{m-1}{d-1}(l-1)^{m-d}(l-3).$$

The first summand counts the cases that do not share the negative value with the query for the virtual attribute. Thus, the d matching attributes must be selected from the $m-1$ observables that are not A_e. Since the query does not have any of the values a_{e1}, a_{e2} for A_e, there are 2 possible attribute values for the other case, namely a_{e1}, a_{e2}. The second summand describes the cases that are identical to the query in A_e. Since this way the query and the case also have the same value for the virtual attribute, they share at least two attribute values. The other $d-2$ attributes must be selected from the remaining $m-1$ observables. The third summand counts the cases that have the same value for the virtual attribute as the query, but are not identical in A_e. The factor $(l-3)$ is due to the fact that for A_e the possible values are restricted to ones that are neither identical to the query's value, and neither a_{e1}, a_{e2}.

The number of cases with similarity d to a query that has one of the values a_{e1}, a_{e2} for A_e is

$$N'_{dis}(d) = \binom{m-1}{d}(l-1)^{m-1-d}(l-2)+$$

$$\binom{m-1}{d-2}(l-1)^{m-d+1} + \binom{m-1}{d-1}(l-1)^{m-d}.$$

The first summand counts the cases that do not share the positive value for the virtual attribute. The factor $(l-2)$ accounts for the fact that the case cannot have a_{e1}, a_{e2} for A_e (otherwise the virtual attribute would be true). The second summand counts those cases that share the same value for A_e. The third summand comprises cases that are not identical in A_e, but share the positive value for A_v.

Another change is that the maximal similarity is increased due to the virtual attribute. Thus we change the accuracy function to

$$A(n) = h_1 \cdot \sum_{d=0}^{m+1} P_{nn}^{d}(d|n,q) \cdot P_f(d) + h_2 \cdot \sum_{d=0}^{m+1} P'^{d}_{nn}(d|n,q) \cdot P'_f(d).$$

The empirical validation (not shown here) resulted again in accuracies that match the formally predicted ones.

5.5 Contextual knowledge

Contextual knowledge specifies the relevance of an attribute in regions of the instance space. In this section we will investigate the effect of contextual knowledge in numerical and nominal instance spaces, in the Japanese Credit Screening domain and for distributed concepts.

5.5.1 Using contextual knowledge to determine relevance of numerical observables

In this section we take a feature-selection (Domingos, 1997) approach, that is, weights are either 1 or 0. This way, one can state that in a certain region of the instance-space an attribute is relevant or irrelevant. The default assumption is that all observables are relevant, if not stated otherwise in a contextual rule. We investigate the benefit of such contextual knowledge on classification with different concepts.
The concepts are defined as follows:

$$f_1(c) \leftarrow (30 \leq A_1(c) \leq 70) \wedge (30 \leq A_2(c) \leq 70)$$

$$f_2(c) \leftarrow (25 \leq A_1(c) \leq 50) \vee [(50 \leq A_1 \leq 75) \wedge$$
$$((A_2 \leq -2 \cdot A_1 + 200) \ \vee \ (A_2 \geq 2 \cdot A_1 - 100)) \ \ ]$$

$$f_3(c) \leftarrow (25 \leq A_1(c) \leq 50) \vee [(50 \leq A_1(c) \leq 75) \wedge (30 \leq A_2(c) \leq 70)]$$

f_1 describes a square region, f_2 describes a region that is a compound of a rectangle and a triangle, and f_3 is a combination of two rectangles (see figure 5.36).
For f_1 we define an inaccurate contextual rule simulating the knowledge that left of the square the attribute A_2 is irrelevant: $relevant(A_2, c) \leftarrow A_1(c) > k$. In the area where A_2 is stated to be irrelevant there is no concept boundary. Since misclassifications occur mostly at concept boundaries, our prediction is that this knowledge will not have much impact. Even more so, also A_1 is irrelevant in that area, so the contextual rule does not seem useful. For completeness' sake, we vary the constant k over the whole range of A_1.
For f_2 we define an inaccurate contextual rule that states that for the left boundary of the rectangle (which spreads over the whole range of A_2) the attribute A_2 is irrelevant: $relevant(A_2, c) \leftarrow A_1(c) > k$. Since the left boundary of the rectangle is formed exclusively by A_1, our prediction is that this

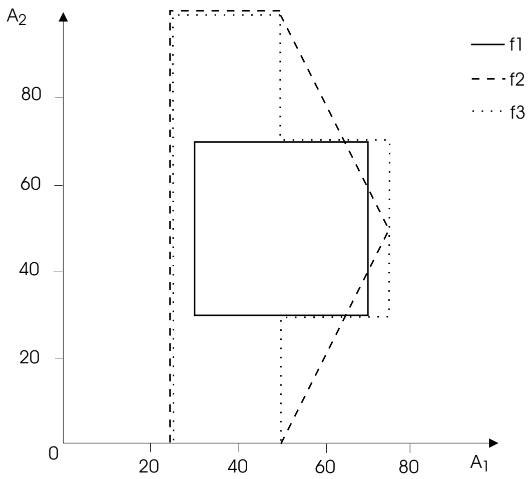

Figure 5.36: The three concepts f_1, f_2, f_3.

contextual knowledge will improve the classification accuracy. Again for completeness' sake, we vary k over the whole range of A_1.

We include f_3 as a contrast to f_2. For the latter, both observables A_1, A_2 are relevant for the triangle region. For f_3, the relevance of A_2 increases with a jump at the right boundary of the left rectangle, as the main distinction between positive and negative cases in that area are the horizontal boundaries of the right rectangle. Thus, f_3 should be more sensitive to inaccurate contextual knowledge describing the irrelevance of A_2 in the area $A_1 > 50$. Again, we vary k in $relevant(A_2, c) \leftarrow A_1(c) > k$.

The accuracies for the different concepts are depicted in figure 5.37.

For all concepts the baseline is the standard similarity measure that uses no contextual knowledge, which is equivalent to setting $k = 0$. As expected, the accuracy increase is smallest for f_1 (around the left concept boundary at $A_1 = 30$). The accuracy for f_1 decreases considerably as the region of purported irrelevance of A_2 overlaps with the concept region. This is not a surprise, because A_2 is relevant for f_1, so the more queries are classified without using A_2, the lower is the accuracy. The small increase at $k = 70$ is due to the fact that the rightmost concept boundary is indeed independent of A_2.

For f_2 the effect of contextual knowledge is better. First of all, the accuracy increases noticeably around $A_1 = 25$ (the leftmost concept boundary) and remains steady while k is within the rectangle-part of the concept ($25 \leq A_1 \leq$

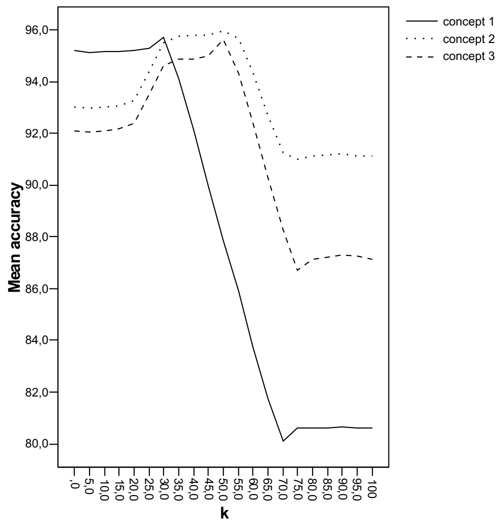

Figure 5.37: Accuracies of similarity measures using contextual knowledge for the concepts f_1, f_2, f_3.

50). This phenomenon is due to the fact that the only concept boundary that is effected is the vertical leftmost one for which A_2 is indeed irrelevant. Only if the other concept boundaries are within the area of believed irrelevance, the accuracy decreases slowly as the area where A_1 is believed to be irrelevant spreads over the triangle part. For the triangle part, A_2 is relevant again, so that the accuracy decrease is not surprising.

The accuracy curve for f_3 is similar to the one of f_2 for values $k < 50$, since the two concepts are equivalent in the region $A_1 < 50$. As expected, the accuracy for f_3 decreases further than for f_2, because A_2 is the only attribute that can separate the cases in the region $50 < A_1 < 75$.

In summary, concept boundaries that are independent of an observable are approximated better if contextual knowledge is used. This knowledge can even be inaccurate, that is, the exact boundaries where the attribute is irrelevant can be stated inaccurately (as long as no other boundaries that depend on the attribute are effected, and as long as the boundary for which the attribute is irrelevant is in the region of purported irrelevance).

We also showed in (Steffens, 2004a) and (Steffens, 2005d) that contextual knowledge for numerical attributes can be useful in similarity-based opponent modelling (refer to chapter 7). In section 6.5 we learn the weights of the

Table 5.3: Accuracy of the knowledge-poor measures for a case-base filled to a quarter, averaged over 5000 runs.

measure	accuracy	standard deviation
standard measure	67.63 %	3.76
without A_2	73.17 %	7.23
without A_3	50.25 %	6.94

relevant attributes and use contextual knowledge to improve weight-learning.

5.5.2 Contextuals for nominal attributes

In order to understand the effect of contextual knowledge on nominal features, we examine a concept that is simple, yet provides several ways to use contextual knowledge. The target concept is

$$f(c) \leftarrow (A_1(c) \neq a_1 \wedge A_3(c) = c_1) \vee (A_1(c) = a_1 \wedge A_2(c) = b_1)$$

in a universe spanned by the attributes $A_1 = \{a_1, a_2, a_3, a_4, a_5\}, A_2 = \{b_1, b_2, b_3, b_4, b_5\}, A_3 = \{c_1, c_2, c_3, c_4, c_5\}$. Obviously, attribute A_2 is only relevant if A_1 has the value a_1, and attribute A_3 is only relevant if A_1 has a value different from a_1.

First of all, the baselines (against which the measure with contextual knowledge has to compete) are the standard similarity measure, the measure that does not use A_2 at all, and the measure that does not use A_3 at all.

The differences in accuracy between the three baselines are highly significant (see table 5.3). Despite the great standard deviation, the baselines are pairwise significantly different from each other ($p < 0.001$ in a two-tailed t-test). Surprisingly, removing A_2 from the similarity measure yields a better accuracy than the standard measure. Obviously, the lack of distinguishing between negative and positive cases when $A_1(c) = a_1$ is more than outweighed by the misclassifications of the standard measure in the other areas where A_2 is used although it is irrelevant. Surprisingly, there is also a difference between the measures that do not use A_2 or A_3. The latter seems to be more important for classification. This is explained by the fact that A_3 defines the concept boundary for more values of A_1 than A_2.

Now, we define rules capturing correct or wrong contextual knowledge. Table 5.4 shows the accuracy of similarity measures using different chunks of

Table 5.4: Accuracy of the measures using contextual knowledge about the relevance of attributes.

measure	accuracy	deviation
correct: $relevant(A_2, c) \leftarrow A_1(c) = a_1$	90.84 %	3.17
wrong: $relevant(A_2, c) \leftarrow A_1(c) = a_2$	79.52 %	4.87
negation of correct: $relevant(A_2, c) \leftarrow A_1(c) \neq a_1$	62.20 %	5.26
correct: $relevant(A_3, c) \leftarrow A_1(c) \neq a_1$	73.78 %	3.42
wrong: $relevant(A_3, c) \leftarrow A_1(c) \neq a_2$	61.67 %	4.90
negation of correct: $relevant(A_3, c) \leftarrow A_1(c) = a_1$	44.02 %	7.12
both correct rules	96.93 %	3.17

contextual knowledge.

Note that for each attribute we tested two kinds of wrong knowledge. One kind just uses a wrong trigger value for the relevance rule. The other kind is the negation of the condition in the correct relevance rule.

Using correct contextual knowledge about A_2 yields higher accuracies than the standard measure or the measure that ignores A_2 (significantly with $p < 0.001$). This is due to the fact that in the knowledge-rich measure, A_2 does not influence the classification when it is irrelevant, but is used when the classification depends on it. Similarly, when using the wrong knowledge that A_2 is relevant if $A_1(c) = a_2$, the accuracy is still better than for the standard measure ($p < 0.001$), because A_2 is not used for cases which have values a_3 or a_4 where A_2 is indeed irrelevant. It is also better than removing A_2 totally ($p < 0.001$), because removing A_2 totally introduces misclassifications if $A_1(q) = a_1$ (where A_2 is relevant). To continue this trend, the measure that incorporates the negation of the correct contextual knowledge performs worst. A_2 is used where it is irrelevant, and not used when it is relevant, increasing the misclassification probability for all cases, no matter what value of A_1 they have.

There is also a difference between attributes. The impact of the correct knowledge about the relevance of A_3 is smaller than of the knowledge concerning A_2. This is no surprise, because the default method to use A_3 fits well with the fact that it is relevant for most cases. Only if a case has value a_1, A_3 should not be used. Thus, the contextual rule is not triggered often. Still, the increase in accuracy is significant ($p < 0.001$).

If the contextual knowledge about A_2 and A_3 is combined, for each case the

relevant attributes are used which leads to high accuracies.
In the next section we apply contextual knowledge to real-world data.

5.5.3 Contextual knowledge in a real-world data set

In this section we exploit the imperfect domain theory of the Japanese Credit Screening domain for contextual knowledge. See figure 5.18 for the structure of the domain theory. We generated rules of contextual knowledge from the domain theory as follows. For each observable it was checked in which subtrees it was used. For each subtree, the literals of the other attributes in that subtree were AND-conjuncted. Finally, if there were several subtrees, the propositions for the subtrees were OR-conjuncted. For example, the observable `number_years` appears as follows in the subtrees for `discredit_bad_region` and `rejected_age_unstable_work` (the domain theory is given in PROLOG notation):

```
discredit_bad_region(Case) :-
    problematic_region(Case),
    number_years(Case,N),
    not(10 < N).

rejected_age_unstable_work(Case) :-
    age(Case, Age),
    59 < Age,
    number_years(Case, N),
    N < 3.
```

Thus, the contextual rule for `number_years` is $relevant(number\_years, c) \leftarrow problematic\_region(c) \vee age(c) > 59$. The contextual knowledge for the other observables was generated analogously. Table 5.5 shows the contextual rules and the associated number of correctly classified cases in a leave-one-out evaluation.
The relevance rule for *gender* (not listed in the table for space constraints) is

$$\begin{aligned} r(gender, c) \leftarrow & jobless(c) \vee item(c) = bike \vee \\ & monthly(c) \cdot months(c) > deposit(c), \end{aligned}$$

and for *job* is

$$r(job, c) \leftarrow male(c) \ \vee \ (female(c) \wedge unmarried(c)) \vee$$

Table 5.5: Accuracy of similarity measures using contextual knowledge for the observables in the JCS domain. The *relevant*-predicate is abbreviated as r due to space constraints.

measure	acc
standard measure	79
$r(unmarried, c) \leftarrow jobless(c) \wedge female(c)$	87
relevance rule for *gender*	82
$r(item, c) \leftarrow female(c)$	82
$r(region, c) \leftarrow number\_years(c) \leq 10$	81
$r(months, c) \leftarrow female(c) \wedge monthly(c) \cdot months(c) > deposit(c)$	80
$r(deposit, c) \leftarrow female(c) \wedge monthly(c) \cdot months(c) > deposit(c)$	79
$r(monthly, c) \leftarrow female(c) \wedge monthly(c) \cdot months(c) > deposit(c)$	79
relevance rule for *job*	78
$r(number\_years, c) \leftarrow region(c) \vee age(c) > 59$	78
$r(age, c) \leftarrow number\_years(c) < 3$	78

$$(female(c) \wedge married(c) \wedge item(c) = bike) \vee$$

$$(female(c) \wedge married(c) \wedge monthly(c) \cdot months(c) > deposit(c)).$$

The effect is a little disappointing. In three situations the accuracy is slightly decreased by adding contextual knowledge. Three times the accuracy remains unaffected. In five situations the accuracy is increased (up to 7% of the 124 test cases).

The results are different depending on the type of the attributes. Except for *job*, using contextual knowledge for nominal attributes $(unmarried, item, region, gender)$ increased accuracy, while for the numerical attributes $(months, deposit, monthly, number\_years, age)$ the accuracies are in the bottom half. The trigger for *job* is only activated by four cases, that is why the effect is so small. Furthermore, for these four cases the predicate *job* partitions well between positive and negative cases, thus the irrelevance prediction is false for this attribute. The bad performance of using contextual knowledge about the numerical attributes is due to the following: The contextual knowledge is correct in the sense that in the respective region the attributes are irrelevant, that is, they do not separate positive from negative cases clearly. However, there exist some intervals, in which positive or negative cases cluster. The standard similarity measure benefits from these clusters, so that the measure with the contextual knowledge performs slightly worse.

In summary, in this domain the contextual knowledge does not decrease accuracy much, but can lead to small increases in accuracy.

5.6 Transformational knowledge

In this section we will investigate how transformational knowledge can be used to better approximate concepts. As described earlier, transformational knowledge is made up of geometrically motivated transformations. There are arbitrary many different affine transformations in geometry, so we focus on the standard operations here, such as translation, reflection, rotation and scaling. Concepts that benefit from such knowledge must also have the corresponding geometrical property.

First of all, let us consider examples of concepts which are prone to be described using geometric transformations. Let us consider a system that records patient data such as temperature, blood pressure etc., for example in order to learn the concept when the patient feels a head-ache. Assume that each data point is associated with a time stamp. Often such time stamps are measured in milli-seconds since a fixed date, resulting in a numerical, open-ended attribute. However, a domain expert might say that the patient state is not a sequence of independent time-points, but that time should be treated as a loop, segmenting the time-line into intervals of 24 hours. This way it is possible to associate the patient states with the time of the day. In other words, a certain temperature at 8am means something different than at 6pm. We will show how the 24h intervals could be translated onto each other, so that the data density is increased.

The above setting is an example for geometrical translation. An example for reflection is the concept ”dangerous forward” from the RoboCup domain. For this concept, a scene where a forward dribbles toward the goal on the left wing is equivalent to a scene where a forward dribbles toward the goal on the right wing. We implemented this as geometrical reflection (Steffens, 2005d, 2005a) (for more details, see chapter 7). Symmetric concepts are rather common in artificial domains (e. g. in board games such as tic-tac-toe (Ragavan & Rendell, 1991)).

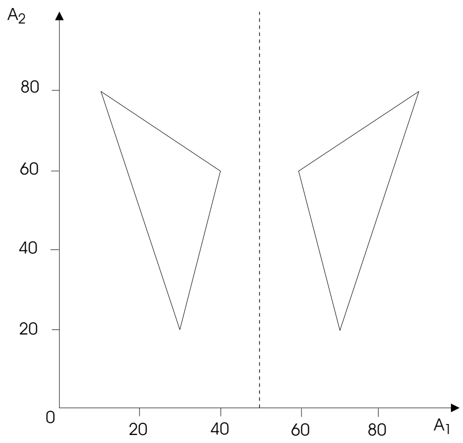

Figure 5.38: The symmetric concept $f(c)$.

5.6.1 Symmetry

Let us now examine symmetry in a less complex domain than simulated soccer. We use a concept that consists of two unconnected triangles in a universe spanned by the numerical attributes A_1, A_2. The triangles are symmetric with respect to the line $A_1 = 50$ (see figure 5.38). Let the predicate $triangle(A_{1,1}, A_{2,1}, A_{1,2}, A_{2,2}, A_{1,3}, A_{2,3}, x, y)$ be true if the point (x, y) is within the triangle defined by the points $(A_{1,1}, A_{2,1}), (A_{1,2}, A_{2,2}), (A_{1,3}, A_{2,3})$. Then the concept is defined as

$$f(c) \leftarrow triangle(10, 80,\ 40, 60,\ 30, 20,\ A_1(c), A_2(c)) \vee$$

$$triangle(90, 80,\ 60, 60,\ 70, 20,\ A_1(c), A_2(c)).$$

In the usual setting of 100 cases in the case-base, 200 test cases, 5000 runs and equal weights, the standard similarity measure yields an average accuracy of 92.02 % (standard deviation 2.28).

The knowledge-rich similarity measure exploits knowledge of the symmetry property: If reflecting the query yields a higher similarity than not reflecting it, than the similarity value of the reflected query to the case is calculated. Otherwise, the non-reflected query is used. Thus, the local similarity function for A_1 is

$$d_1(c_n, c_m) = max\left(1 - \left(\frac{|X_n - X_m|}{range(A_1)}\right)^2, 1 - \left(\frac{|(100 - X_n) - X_m|}{range(A_1)}\right)^2\right),$$

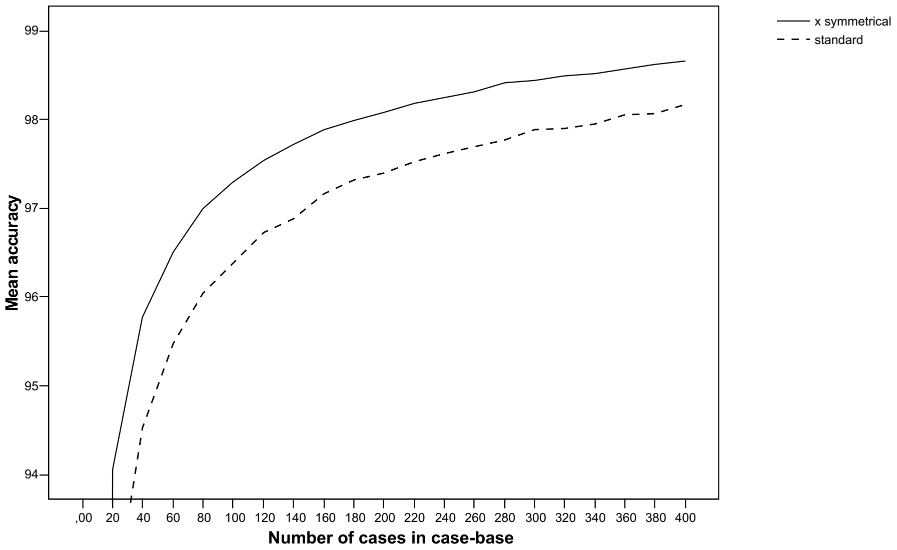

Figure 5.39: Accuracies of the standard measure and the measure exploiting symmetry. The accuracy values of the knowledge-rich measure correspond to the values of the standard measure at the doubled case-base size.

where $X_n = A_1(c_n)$ and $X_m = A_1(c_m)$. This measure yields an average accuracy of 94.17 % (standard deviation 1.98). This value is almost identical to the accuracy of the standard measure if the case-base size is doubled: 94.18 % (standard deviation 1.77). Furthermore, plotting the accuracy of the standard measure and the knowledge-rich measure (see 5.39) suggests that the knowledge-rich measure yields the same accuracy values as the standard measure with the doubled case-base size.

This suggests that using symmetrical knowledge can simulate the effect of increasing the case-base size (provided the concept has the symmetry property). The similarity between the query and a reflected case can be as high as between the query and a non-reflected nearby case. Thus, all cases of the instance space are treated as if they were on the same side of the mirror line as the query. This results in a simulated increase of the case-base size proportional to the smaller area of the mirrored partitions.

If the instance space does not have a (fully) symmetric structure, reflection will result in mixing positive and negative cases in the areas that are not symmetric. Consider the instance space depicted in figure 5.40. The two areas A and B are not symmetric. Nevertheless, if the similarity measure

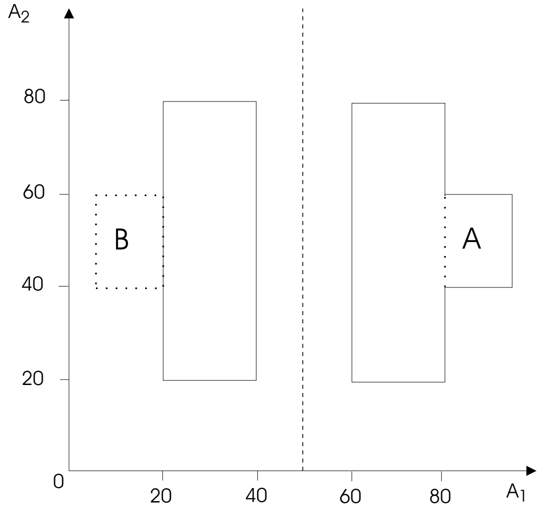

Figure 5.40: A concept that is not fully symmetrical to $A_1 = 50$. Concept boundaries are depicted by solid lines, imaginary auxiliary lines are depicted by dashed lines. Thus, region A belongs to the right concept area, whereas B (the reflection of A) does not belong to the concept area.

assumes that A_1 is symmetric wrt. to $A_1 = 50$, then cases from A will be used to classify queries in B, and vice versa. Assuming a uniform distribution queries in A and B will be classified wrongly with a probability of 50% [4]. Misclassifications do not only occur at the concept boundaries, but can occur all over the nonsymmetric areas. Thus, the additional error probability will increase proportionally with the size of the nonsymmetric areas. The relation between non-symmetric areas and accuracy of a measure assuming symmetry is illustrated by figure 5.41. For the concept $f(c) \leftarrow 30 \leq A_1(c) \leq k \wedge 30 \leq A_2(c) \wedge 70$ the left boundary is varied. The concept is fully symmetrical wrt. $A_1 = 50$ for $k = 70$, and the size of the non-symmetric areas is $|70 - k| \cdot 40$. The accuracy is roughly linear, supporting the proportionality hypothesis. However, some noise is introduced by the normal misclassifications that occur at the concept boundaries independently of the symmetry assumption.

[4] This is true if we ignore interferences at the concept boundaries. The concept $f(c) \leftarrow A_1(c) < 50$ has a theoretical accuracy of 50% for the measure using the symmetry assumption independent of the case-base size. Empirically for a case-base of 100 cases, 200 test cases and 5000 runs, the accuracy is 49.65% which supports this hypothesis.

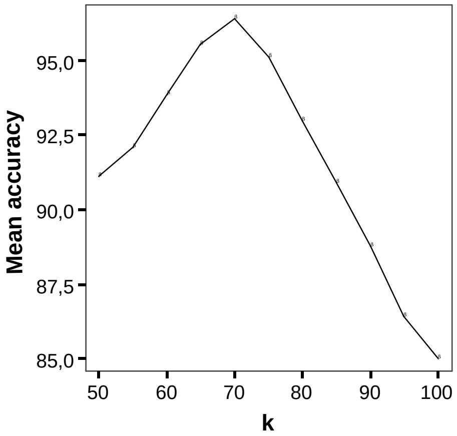

Figure 5.41: Accuracies of the similarity measure that assumes that the concept is symmetrical to $A_1 = 50$. The concept is $f(c) \leftarrow 30 \leq A_1(c) \leq k \wedge 30 \leq A_2(c) \wedge 70$.

5.6.2 Rotation

The only concepts that are globally invariant under rotation are spherical. It is possible to combine rotational knowledge with contextual triggers so that the concept is invariant under rotation only in certain subregions, but we consider only global transformational knowledge here. Circular concepts are rather uncommon in similarity-based classification, but it seems feasible to apply similarity measures with rotational knowledge if it is known that the concept is circular around a certain point, but the diameter is unknown. Since this set-up seems rather uncommon, we restrict our analysis to a short investigation of a concept that has the shape of a circle around the point $50, 50$ with the diameter 40. In our usual experimental setting, the standard similarity measure yields an average accuracy of 96.05 % (standard deviation 1.61), and the measure that applies rotational knowledge by comparing cases via their distance to the concept center yields 99.51 % (standard deviation 0.70). The difference is highly significant ($p < 0.001$).

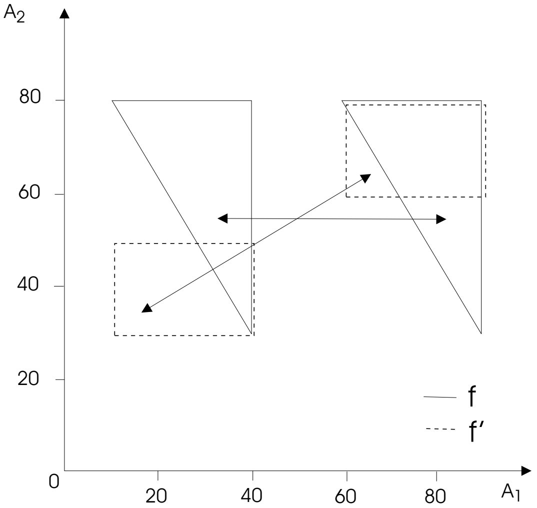

Figure 5.42: f and f' are invariant under translation.

5.6.3 Translation

Translation knowledge seems to be applicable to real-world concepts as we have illustrated with the time-attribute in the patient data example. In the following experiments we will show that knowledge about the translational invariance of a concept can increase classification accuracy, but only if the instance space can be partitioned into congruent regions. This means, that these congruent regions are not allowed to overlap and that the translation must map the regions onto each other. We will also examine the result of using translations in the similarity measure if the instance space is not partitioned into congruent regions.

But first we investigate concepts that have a translational structure. The concept

$$f(c) \leftarrow triangle(10,80,\ 40,80,\ 40,30,\ A_1(c), A_2(c)) \vee$$

$$triangle(60,80,\ 90,80,\ 90,30,\ A_1(c), A_2(c))$$

(see figure 5.42) is invariant under translation of 50 on the A_1-dimension (in our implementation, cases with $A_1(c) < 50$ are translated by 50, and by -50 else). The two halves generated by the line $A_1(c) = 50$ are congruent to each other. With the usual setting, the standard measure yields an average accuracy of 91.83 % (standard deviation 2.33).

The knowledge-rich similarity measure which uses transformational knowledge uses the local similarity function

$$d_1(c_n, c_m) = \begin{cases} max\left(1 - \left(\frac{|X_n - X_m|}{range(A_1)}\right)^2, 1 - \left(\frac{|X_n + 50 - X_m|}{range(A_1)}\right)^2\right) & : \ if\ X_n \leq 50 \\ max\left(1 - \left(\frac{|X_n - X_m|}{range(A_1)}\right)^2, 1 - \left(\frac{|X_n - 50 - X_m|}{range(A_1)}\right)^2\right) & : \ if\ X_n > 50 \end{cases}$$

where $X_n = A_1(c_n)$ and $X_m = A_1(c_m)$. This measure yields an accuracy of 94.18 % (standard deviation 1.93). This is a significant increase. Furthermore, it is equivalent to the accuracy of the standard measure on a case-base with the doubled size (94.13 %, standard deviation 1.84). Again, this is evidence for the fact that transformational knowledge simulates increasing the case-base density. Since the instance space is partitioned into two halves which are translated onto each other, the effect is that of a case-base with the doubled size.

Translation can of course also be done on more than one dimension. The concept

$$f'(c) \leftarrow (10 \leq A_1(c) \leq 40 \wedge 30 \leq A_2(c) \leq 50) \vee$$
$$(60 \leq A_1(c) \leq 90 \wedge 60 \leq A_2(c) \leq 80)$$

(see figure 5.42) is invariant under translation by $\overrightarrow{t} = (50, 30)$. The average accuracy of the standard measure is 93.86 % (standard deviation 2.11). The measure which exploits translational knowledge returns the maximum of the standard similarity on the original query and the case, of the standard similarity of the case and the query translated by $(50, 30)$, and of the standard similarity of the case and the query translated by $(-50, -30)$. This increased computational effort yields an average accuracy of 95.70 % (standard deviation 1.66), which is again equivalent to the accuracy of the standard measure on a case-base with the double size (95.67 %, standard deviation 1.56).

However, if the instance-space cannot be partitioned into congruent regions wrt. the translation, using translations decreases the accuracy, even if the boundaries of the distributed concept parts are shifted correctly onto each other. This is illustrated by the concept

$$f''(c) \leftarrow triangle(10, 80,\ 30, 80,\ 60, 20,\ A_1(c), A_2(c)) \vee$$
$$triangle(40, 80,\ 60, 80,\ 90, 20,\ A_1(c), A_2(c))$$

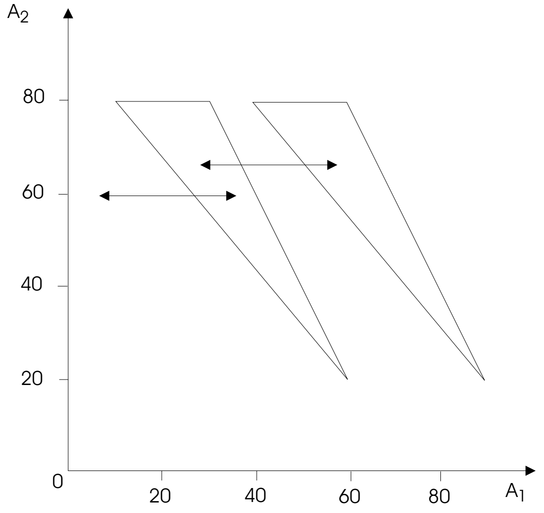

Figure 5.43: The instance space for f'' cannot be partitioned into congruent regions wrt. the translation (30,0). For example, the point (5,60) which does not belong to the concept, is translated to (35,60), which is inside the concept area of the left triangle.

(see figure 5.43). Although the two triangles can be translated onto each other by 30 on the A_1 dimension, the instance space cannot be partitioned into congruent regions wrt. the translation. Thus, the query $(5, 60)$ which is not a concept member, will be translated to $(35, 60)$, which is in the left triangular concept region. This will lead to a misclassification. Accordingly, the average accuracy of the standard measure (91.19 %, standard deviation 2.39) is better than the accuracy of the measure that makes use of the translation (88.47 %, standard deviation 2.59). With the same considerations as for symmetry, the additional error probability increases with the size of the non-congruent areas. So, although there are local regions that can be translated onto each other, global translation decreases the accuracy. To remedy this, a trigger is required that activates the transformation only if the query is in a certain region. This can easily be incorporated into a similarity measure, but we will not investigate triggers further.

5.6.4 Scaling

Concepts that are invariant under scaling are certainly only minimally relevant in real-world applications. But for completeness' sake we will investigate such concepts briefly. The concept $f(c) \leftarrow 50 \leq A_2(c) \leq 70$ is invariant under scaling of the A_1 dimension. The standard measure results in an average accuracy of 93.47 % (standard deviation 2.10). The extended similarity measure uses the local similarity function

$$d_1(c_n, c_m) = \begin{cases} max\left(1 - \left(\frac{|X_n - X_m|}{range(A_1)}\right)^2, 1 - \left(\frac{|X_n \cdot 1.5 - X_m|}{range(A_1)}\right)^2\right) & : \; if X_n \leq X_m \\ max\left(1 - \left(\frac{|X_n - X_m|}{range(A_1)}\right)^2, 1 - \left(\frac{|X_n / 1.5 - X_m|}{range(A_1)}\right)^2\right) & : \; else \end{cases}$$

where $X_n = A_1(c_n)$ and $X_m = A_1(c_m)$. We determined the scaling factor of 1.5 empirically for this concept, as for the geometrical structure the scaling factor is arbitrary. This extended measure achieves an average accuracy of 95.63 % (standard deviation 1.65). This is significantly better than the standard measure with the same case-base, and also significantly better (although only slightly) than using the standard measure on a case-base with the doubled size (95.38 %, standard deviation 1.61).

5.6.5 Incorporating transformational knowledge as virtual attribute

Transformational knowledge can also be incorporated as virtual attribute. That is, the transformation is applied to case and query, and the transformed values are treated as additional attributes. However, this incorporation method has no effect on the classification behavior. In our experiments with the above concepts, the virtual attribute method results in accuracies that are not significantly different from the standard measure. Even adapting the weights of the virtual attributes has no effect.

The reason is that virtual attributes do not change the topology of the instance space. Assume that case c is very similar to query q according to the standard similarity measure. Then also the virtual attributes (i. e. the transformed values) of c will be very similar to q (at least for standard transformations such as translation, scaling, rotation and reflection). In contrast, assume a second case c' is less similar to q than c is, but corresponds to q after transformation (i. e. $trans(c') = q$). Even though the virtual attributes of c' may have similar values to q, the original case attributes will still be different, so that the similarity of q to c will be higher than of q to c'. In short, cases that are highly similar to the query according to the standard measure will also be highly similar according to the measure that uses transformed attributes as additional virtual attributes.

Thus, for transformational knowledge the virtual attribute method does not change the accuracy.

5.6.6 Conclusion for transformational knowledge

We investigated the effect of knowledge about the invariance of concepts under geometrical standard operations. The result is that transformational knowledge can increase the classification accuracy similar to increasing the case-base size. This is consistent with the finding of (Ragavan & Rendell, 1991), who showed that the worth of symmetry knowledge for inductive learning can be stated as number of additional training cases. Our results suggest that their finding also holds for similarity-based classification and other transformations.

Our analysis also showed that using the transformed values as additional virtual attributes does not change the classification behavior of the classifier, because the neighborhood-relations remain unchanged.

5.7 Conclusion

In this chapter we have investigated the impact of the different knowledge types on classification accuracy. We have implemented the incorporation methods and have systematically tested them in artificial and real-world domains. For some methods we have also given a formal analysis.

While in most other knowledge-rich CBR approaches a complete or correct knowledge base is required (e. g. (Boerner, 1994)), we have examined the consequences of inaccurate, partial, and inconsistent knowledge. The conclusion is that partial (but otherwise correct) knowledge increases classification accuracy. Especially if several knowledge chunks are incorporated into the similarity measure, the increase in classification accuracy is substantial. In contrast, the benefit of inaccurate or inconsistent knowledge is more constrained, since the knowledge is only useful if the inaccuracy and inconsistency are not too great. The virtual attribute method turned out to be more robust against imperfect knowledge than the more specialized incorporation methods, since it keeps the observables unchanged as basis for comparison. Only for transformational knowledge, the virtual attribute method is not suited because it does not change the classification behavior.

In summary, imperfect knowledge can increase the classification accuracy of similarity measures. These results form a new incentive to exploit knowledge even in domains where no perfect knowledge is available. In this sense it refutes the traditional "knowledge-poor" approach of similarity-based classification which does not consider domain knowledge. Since the requirements for knowledge in terms of correctness and completeness are softened, knowledge acquisition from domain experts (which usually results in inaccurate and incomplete knowledge) seems viable. We believe that our results are a step to facilitate the interview process, because even knowledge that cannot be acquired perfectly can be useful.

Chapter 6

Weighting attributes

In this chapter we apply weight-learning methods to deal with incomplete domain theories. The relevance of virtual attributes for the classification goal is estimated by adjusting their weights. Furthermore we extend existing weight-learning algorithms to exploit domain knowledge.

6.1 Introduction

In the previous chapters we made the following two simplifications: First of all, all attributes were weighted equally. Secondly, if we used partial knowledge it was known to be relevant. In this chapter we tackle both of these issues by using weight-learning algorithms. This way, the first issue is remedied because attributes receive weights that are correlated with their relevance. Relevant attributes will be associated with high weights, irrelevant attributes with low ones. The second issue can be characterized as follows. If a domain theory is incomplete, there might be intermediate concepts which are completely defined but whose relation to the classification goal is unknown. In section 3.5.1 we called this phenomenon ”gaps at the top” of the domain theory. A consequence of such gaps is that it is uncertain whether an intermediate should be added as virtual attribute for the classification goal or not. Weight-learning methods are a means to use the case-base in order to learn whether an intermediate will be a useful virtual attribute or not. Learning weights has the additional benefit that the relevance of virtual attributes can be graded. Even if two virtual attributes are relevant, it is possible that one is more relevant than the other. Such information can be

learnt with weight-learning algorithms.
In the next section we motivate why intermediate concepts need to be selected carefully and that weight-learning is an appropriate means.

6.2 The need to select intermediate concepts

We reported experimental results that suggested that not all intermediate concepts in the JCS domain theory are good virtual attributes (section 5.2.5). Thus, a criterion for selecting intermediate concepts is needed.
Since a domain theory is available, it seems promising to analyze the structure of the theory in order to identify concepts that will form good virtual attributes. If the impact of an intermediate concept as virtual attribute can be inferred from the domain theory, it would be unnecessary to process training cases. Thus, a methodology to exploit the structure of the domain theory seems promising. Unfortunately, we did not find any cross-domain theory characteristics that predict the impact of intermediate concepts.
For example, in the JCS domain theory we calculated the following characteristics for each intermediate concept:

- Level of abstraction: We operationalize abstraction as number of inference steps from the observables. For example, the concept *bad_credit* (see figure 5.18) does not use observables but other intermediates and has a level of abstraction of 2. The intermediate $unmatch\_female$ is directly related to observables and has a level of abstraction of 1. Our hypothesis was that concepts that are abstract and thus close to the classification goal will be good virtual attributes.

- Connectivity: The number of concepts that use an attribute describe the attribute's connectivity. We hypothesized that concepts that are used by many other concepts are highly relevant and will thus be good virtual attributes.

- Specificity: The specificity of a concept is operationalized as the number of its conditions. We predicted that highly specific intermediates perform bad as virtual attributes because they are only relevant in small regions.

Table 6.1 shows the characteristics for each intermediate concept from the JCS domain theory.

Table 6.1: Characteristics of intermediate concepts from the JCS domain theory, and the accuracy that a similarity measure obtains if the intermediate is added as virtual attribute (weighted with 10) to the standard measure.

intermediate	abstractness	connectivity	specificity	acc
jobless_male	1	1	2	74
jobless_unmarried_female	1	1	3	70
unmatch_female	1	2	2	77
discredit_bad_region	1	1	2	76
rejected_age_unstable_work	1	1	2	75
bad_credit	2	1	5	81
ok_credit	3	0	1	81

In earlier work we correlated these characteristics to the accuracy of similarity measures that contained exclusively one intermediate attribute (and not any observables) (Steffens, 2004b). Unfortunately, these characteristics do not predict the accuracy of similarity measures that use a virtual attribute and observables. Although our hypothesis that abstract intermediates are good, is not contradicted, it is not strongly supported by these sparse data either. The two intermediates with the highest abstractness also yield the highest accuracy. Furthermore, there is no difference in accuracy between *bad_credit* and *ok_credit*, although the latter is more abstract. Our connectivity-hypothesis is contradicted, as there is no clear trend visible. Also the specificity-hypothesis is contradicted, because the intermediate with the highest specificity yields the highest accuracy (just as the one with the lowest specificity).

Now, it may be argued that we just did not use the correct characteristics or that we need more data. To make sure that analysis of the structure of domain theories cannot predict the impact of intermediate concepts, we investigated another domain from the UCI Machine Learning Repository. The Mechanical Analysis domain provides a complex domain theory (see figure 6.1).

As can be seen, the intermediate concepts *v_alta* and *v_molto_alta* have identical structures and only differ in their name. Both are defined using *Cpm* and *Mis*, and are used for defining *v_pericolosa*. If the structure of the domain theory is a predictor of the impact of an intermediate as virtual attribute, the accuracy of these two intermediates should be equal. However, when adding *v_molto_alta* as virtual attribute to the standard measure, 2595 cases are classified correctly, and when adding *v_alta*, 2615 cases are classified cor-

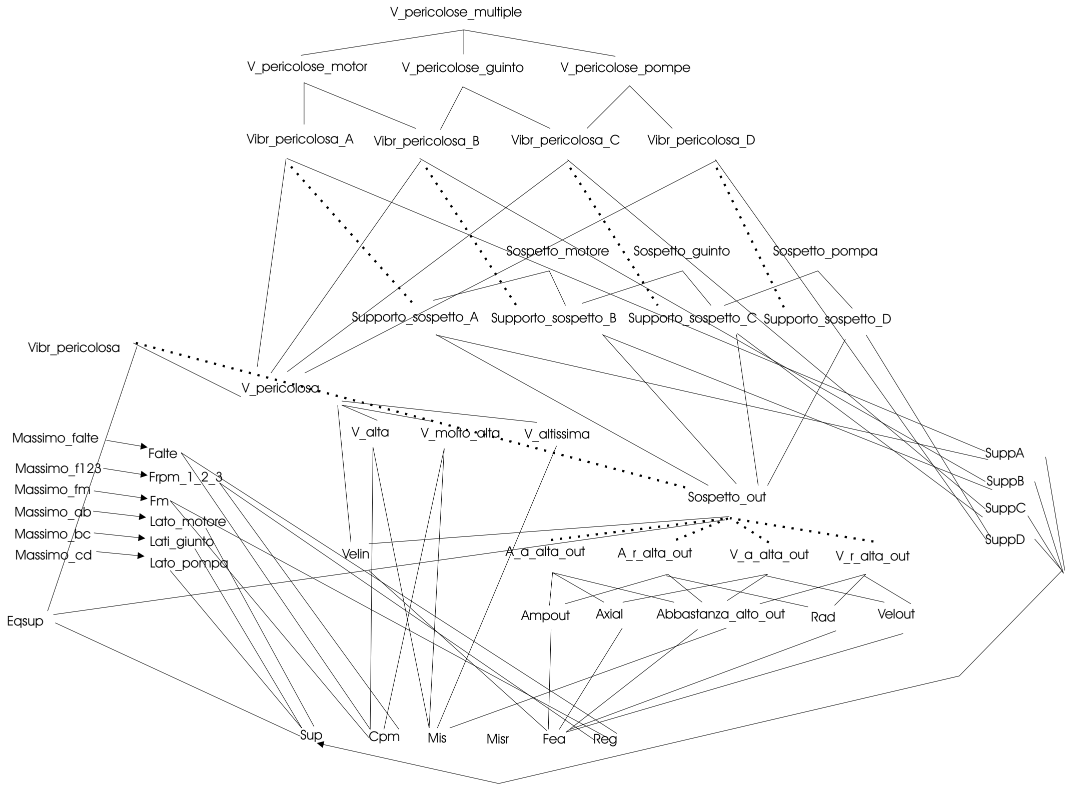

Figure 6.1: The structure of the domain theory of the Mechanical Analysis domain. Nodes denote attributes, arcs denote relations. (The labels are Italian.)

rectly. The former yields the same result as the standard measure, and the latter increases accuracy. Thus, we think that the structure of the domain theory cannot be used to determine whether an intermediate concept will be a good virtual attribute.

In the literature there is not much work on how to decide whether an additional attribute will improve a similarity measure. The existing literature does not cope with the idea of extending similarity measures, but is rather focussed on how to reduce the number of attributes while retaining classification accuracy (Wettschereck et al., 1997). Thus, in the next section we apply weight-learning methods in the following way: We add all available intermediate concepts to the similarity measure and let the weight-learning methods filter irrelevant attributes by converging on small weights. Note that the weight-learning process is not an additional effort that is caused by the use of virtual attributes. Instead, the weights of observables have to be learnt anyway, so that the virtual attributes can be handled in the same process. After that we analyze how weights can be used to filter bad inaccurate virtual attributes.

6.3 Learning weights in the JCS and PGS domain

In this section we apply weight-learning methods in the Japanese Credit Screening and the Promoter Gene Sequences domain. We show that learning weights of virtual and observable attributes increases the classification accuracies.

Weighting methods

In order to approximate the relevance of intermediate concepts and observables in the JCS and PGS domain, we implemented several weight-learning methods that are described in the literature. According to the classification of weighting methods as proposed in (Wettschereck et al., 1997), we selected four methods with performance bias, and six with preset bias (i. e., statistical and information-theoretic methods).

- Performance bias: Weighting methods with a performance bias classify instances in a hill-climbing fashion. They update weights based on the outcome of the classification process. The performance bias performs well if there are many irrelevant features (Wettschereck et al., 1997). Since the intermediate concepts of the domain theories can be assumed to be relevant, we expected performance bias methods to perform badly.

1. EACH (Salzberg, 1991) retrieves the most similar case to a query. If the case has the same class as the query, the weight of matching features are increased and the weight of mismatching features are decreased by a hand-coded value. If the case has a wrong class, the weights of matching features are decreased, and weights of mismatching features are increased.
2. IB4 (Aha, 1992) is a parameter-free extension of EACH. It makes use of the concept distribution and is thus sensitive to skewed concept distributions. It assumes that the values of irrelevant features are uniformly distributed.
3. RELIEF (Kira & Rendell, 1992) is a feature selection- rather than feature weighting-algorithm. It calculates weights based on the instance's most similar neighbors of each class and then filters attributes whose weights are below a hand-coded threshold.
4. ISAC (Bonzano et al., 1997) increases weights of matching attributes and decreases weights of mismatching attributes by a value that is calculated from the ratio of the prior use of the instance. The more often the instance was retrieved for correct classifications, the higher the update value.

- Preset bias: The bias of the following methods is based on probabilistic or information-theoretic concepts. They process each training instance exactly once.

  1. CCF (Creecy, Masand, Smith, & Waltz, 1992) binarizes attributes and weights them according to the target classes' conditional probability given a feature.
  2. PCF (Creecy et al., 1992) is an extension of CCF which takes the distribution of the feature's values over classes into account. It calculates different weights for different classes.
  3. MI (Daelemans & Bosch, 1992) (for mutual information) calculates the reduction of entropy in the class distribution by attributes and uses it as the attribute weight.

4. CD (Nunez et al., 2002) creates a correlation matrix of the discretized attributes and the classes. The weight of an attribute increases with the accuracy of the prediction from attribute value to class.
5. VD (Nunez et al., 2002) extends CD in that it considers both the best prediction for a class and the predictions of all attributes.
6. CVD (Nunez et al., 2002) combines CD and VD.

Results

All available intermediate concepts were added as virtual attributes to the similarity measure, and then the weights of observables and virtual attributes were learnt. Also for the standard measure the weights of the observables were learnt. For evaluation we used the leave-one-out method. For most of the weighting methods, the knowledge-rich similarity measure performs better than the standard one. In table 6.2 we underline the accuracy of the knowledge-rich similarity measure if it outperformed the standard similarity measure when using the same weighting method. In the PGS domain, seven of ten weighting methods perform better if the similarity measure is extended with virtual attributes. Even more so, in the JCS domain the accuracies of eight of ten weighting methods were improved by using virtual attributes.

In its optimal setting, with an accuracy of 98.11% our approach performs also better than the results from the literature reported for the PGS domain. The accuracy of KBANN in (Towell, Shavlik, & Noordenier, 1990) is 96.23%, which to our knowledge was the highest accuracy reported so far and also used the leave-one-out evaluation. Note that KBANN also uses domain knowledge which is encoded into the neural network. We found no classification accuracy results for JCS in the literature[1].

Obviously, these improvements are not restricted to a certain class of weighting methods. For example, methods with performance bias (most notably ISAC), information-theoretic bias (i. e. MI), and with a statistical correlation bias (e. g. VD) benefit from processing virtual attributes.

Even in the PGS domain, the improvements are substantial. This is surprising, since the domain knowledge is the worst possible since it classifies at chance level when used for rule-based classification. This is a promising

[1] The domain often referred to as 'credit screening' with 690 instances is actually the credit card application domain.

Table 6.2: Classification accuracies of the standard similarity measure without (w/o) virtual attributes and the measure with (w/) virtual attributes. The columns report the accuracies for the unweighted classification and for several weighting methods.

Domain	unw.	EACH	RELIEF	IB4	ISAC	CCF	PCF	MI	CD	VD	CVD
JCS (w/o)	74.19	74.19	78.23	74.19	72.58	72.58	72.58	74.19	74.19	72.58	71.77
JCS (w/)	74.19	72.58	79.03	72.58	79.03	73.39	75.0	75.0	77.42	75.0	75.0
PGS (w/o)	86.79	89.62	96.23	88.68	50.0	85.85	87.74	68.87	88.68	77.36	83.02
PGS (w/)	85.85	93.40	96.23	90.57	96.23	91.51	86.79	98.11	88.68	97.17	87.74

result as it shows that adding intermediate concepts may increase accuracy even if the domain theory is very inaccurate.

6.4 Weight-learning for inaccurate virtual attributes

The experiments in the JCS domain have the limitation that there were no virtual attributes that deteriorated performance. The worst virtual attributes achieved the same accuracy as the standard measure (refer to figure 5.19). In this section we investigate whether weight-learning methods can be used to deal with inaccurate virtual attributes so that bad virtual attributes are filtered away. First we give a formal analysis about the probability that a weighted virtual attribute has no effect on the classification. By doing this, we can understand how small the weights have to be for filtering out a bad virtual attribute. Then we run weight-learning experiments in order to check whether learning algorithms can balance the influence of inaccurate virtual attributes. That is, based on the curve in figure 5.3 we examine the following virtual attributes:

- virtual attributes that make the measure perform similar to the standard measure,
- virtual attributes that make the measure perform worse than the standard measure,
- virtual attributes that make the measure perform better than the standard measure,
- correct virtual attributes.

6.4.1 Formal analysis

First we analyze formally how small the weight of a binary virtual attribute has to be in order to cancel out the effect of the virtual attribute. Assume the virtual attribute describes a concept boundary at $A_1 = k$ that partitions a two-dimensional instance-space into two parts. Let there be a similarity measure s that uses this virtual attribute $A_v(c) \leftarrow A_1(c) > k$. We have reported earlier that for weights that are great enough, the virtual attribute makes misclassifications at the concept boundary disappear. Now we want to analyze how small the weight has to be so that the effect of the virtual attribute vanishes. This weight threshold cannot be an absolute threshold, but can only be described probabilistically, since it depends on the case-base distribution.

In order to analyze the weight of the virtual attribute A_v, we look at situations where a case, that has a different value for A_v than the query, is the nearest neighbor of the query. That is, the case is more similar to the query than all other cases which share the value for A_v with the query: $\exists c_1, q, \forall c_2 : s(c_1, q) > s(c_2, q) \wedge A_v(c_1) \neq A_v(q) \wedge A_v(c_2) = A_v(q)$ (refer to figure 6.2).

Obviously, c_1 and q have a local similarity of 0 for A_v (i. e. $d_v(c_1, q) = 0$), while c_2 and q have a local similarity of 1 (i. e. $d_v(c_2, q) = 1$). c_1 has to be so much spatially closer to the query than c_2 is, that it can overcome the factor of the virtual attribute, which amounts to $w_v * 1$, where w_v is the weight of A_v.

Consider the line of equal similarity around q on which c_1 lies. Since A_1 and A_2 can be weighted differently, this line is an ellipse. Cases on the other side of the line $A_1 = k$ are at least as similar to q as c_1 is, if they are within an ellipse that is extended by $w_v * 1$. This is due to the fact that these cases can

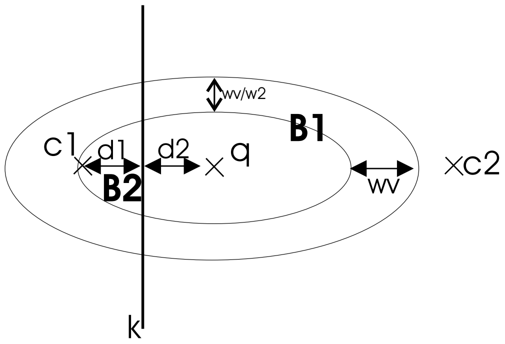

Figure 6.2: Situation where a case c_1 that is on the other side of a separating line (as induced by a virtual attribute) is more similar to the query q than a case c_2 on the same side. Without loss of generality (since only the relative weights are important), w_1 is assumed to be 1. w_v is the weight of the virtual attribute.

accommodate greater spatial distance with their shared value of A_v. Thus, c_1 is the nearest neighbor of q iff there is no case in area B_1 (depicted in figure 6.2), where B_1 is the part of the extended ellipse around q which is on the same side of the separating line as the query. B_2 is the part of the smaller, unextended ellipse on the other side of the separating line.
Let p_1 be the probability that there is no case in B_1, and p_2 be the probability that at least one case is in B_2 (S and λ are defined as in section 5.2.1):

$$p_1 = \left(1 - \frac{B_1}{S}\right)^{S*\lambda}, \; p_2 = 1 - \left(1 - \frac{B_2}{S}\right)^{S*\lambda}$$

The areas of B_1 and B_2 can be calculated as follows:

$$B_1 = \pi * (d_1 + d_2 + w_v) * \frac{d_1 + d_2 + w_v}{w_2} - \frac{2 * \frac{d_1+d_2+w_v}{w_2}}{(d_1 + d_2 + w_v)} * \int_{d_2}^{d_1+d_2+w_v} \sqrt{(d_1 + d_2 + w_v)^2 - x^2}$$

$$B_2 = 2 * \frac{\frac{d_1+d_2}{w_2}}{d_1 + d_2} * \int_{d_2}^{d_1+d_2} \sqrt{(d_1 + d_2)^2 - x^2}$$

Then, the probability that B_1 is empty and at least one case is in B_2 is

$$P = \int_0^\infty \int_0^\infty p_1 * p_2 \; dd_1 \; dd_2.$$

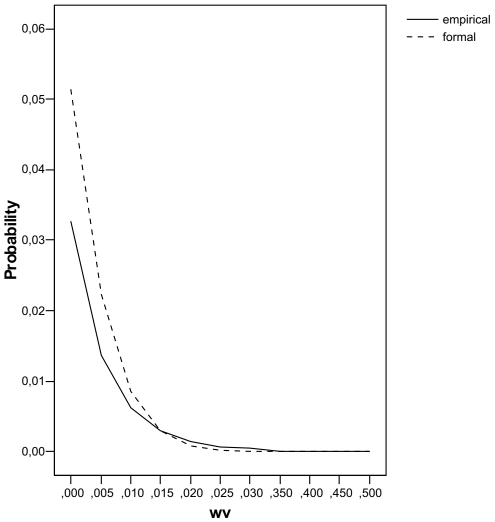

Figure 6.3: The curves show the formal and empirical probability that a case, that is on the other side of a separating line than the query, is the nearest neighbor to the query.

To validate this formal analysis, we compare the prediction to empirical results. We set $k = 50$, let A_1 and A_2 be in the interval $[0, 100]$, have 100 cases in the case-base, 1000 test cases, and run the experiment over 10000 runs. The formal prediction and the empirical results are quite similar (see figure 6.3).

Both the formal and the empirical curve show that the probability (that a case on the other side of the separating line is the nearest neighbor) is very small in general and goes to almost 0 already for small values of w_v. That means that the effect of a virtual attribute is only cancelled if its weight is very small. In the next section we test empirically whether existing weight-learning methods can converge to such low weights.

Also the virtual attributes in real-world data behave like our analysis suggests. We add intermediates from the JCS domain theory to the similarity measure and vary its weight. The curve depicted in figure 6.4 shows the same low sensitivity of virtual attributes to weights. Only if the weight of the beneficial virtual attribute is small, the errors increase. The accuracy does not change anymore if the weight is above a certain threshold.

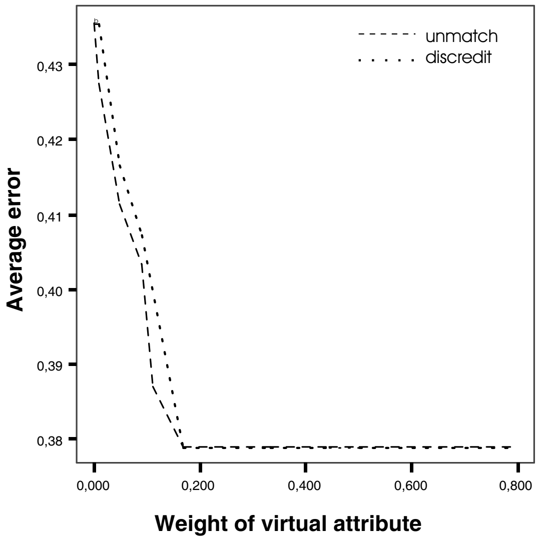

Figure 6.4: The curves show the error that a measure using the virtual attribute *unmatch_female* or *discredit_bad_region* makes if the virtual attribute is weighted as plotted on the horizontal axis.

6.4.2 Weight-learning experiments with bad inaccurate virtual attributes

In the following experiment we test whether weight-learning algorithms are sensitive enough to learn small weights for bad inaccurate virtual attributes, and high values for good inaccurate virtual attributes.

We performed experiments in a universe spanned by three numerical attributes. As target concept we chose a centered cuboid $f(c) \leftarrow 20 \leq A_1(c) \leq 80 \wedge 20 \leq A_2(c) \leq 80 \wedge 20 \leq A_3(c) \leq 80$. The similarity measure uses a virtual attribute of the form $A_v(c) \leftarrow A_1(c) > k$, where k is chosen as described below. We chose such a regular target concept, because the optimal weight setting of the observables is to use equal weights. Thus, if there is an increase of accuracy from the similarity measure, which uses equal weights for all observables and the virtual attribute, as compared to the similarity measure that uses learnt weights, this increase must be due to the weight of the virtual attribute.

Four different values for k were used, namely 0,10,17 and 20. These values were chosen because they describe virtual attributes that lead to

Table 6.3: Accuracies of the similarity measure using equal weights or learnt weights.

k	equal	RELIEF	RELIEF weight	IB4	IB4 weight
0 (baseline)	87.36	87.23	0	87.30	0.26
10	86.97	86.95	0.13	87.07	0.23
17	87.85	87.77	0.31	87.75	0.24
20	88.82	88.88	0.43	88.86	0.25

- the same behavior as the standard measure ($k = 0$)
- decreased performance ($k = 10$)
- slightly increased performance ($k = 17$)
- or increased performance ($k = 20$).

Our hypothesis was that the virtual attributes that used $k = 20$ would receive the highest weights, followed by the virtual attribute that uses $k = 17$. Finally, we hoped that the attribute that used $k = 10$ would receive such a low weight that its deteriorating effect vanishes. Table 6.3 shows the accuracies for a case-base of 100 cases, 200 test cases, and 5000 runs. We ran experiments with the weight-learning methods RELIEF and IB4, because they are well-suited for numerical attributes.

The accuracies obtained by RELIEF and IB4 are not significantly better than the accuracies obtained with equal weights. However, the weights for the virtual attribute that were learnt by RELIEF support our hypothesis from above. The virtual attribute that has a deteriorating effect on accuracy receives lower weights than the virtual attributes that increase accuracy. For RELIEF, there is a clear trend that good virtual attributes receive higher weights. This trend can also be seen in the IB4 weights, but the differences are so small that they might be due to noise.

Remember that the weight has to be very small in order to cancel the effect of a bad virtual attribute. Although the learnt weight for the bad virtual attribute is already the smallest, it is still not small enough to make the deteriorating effect disappear. It might be the case that the learning methods need more training instances in order to converge to such low weights. Thus, we increased the case-base size to 1000 and tested the virtual attribute for $k = 10$. Using learnt weights, the accuracy is not significantly higher (94.20%

for RELIEF and 94.24% for IB4) than when using equal weights (94.25%). On average, the learnt weight for the virtual attribute was 0.21 for RELIEF and 0.24 for IB4 which are even greater than the weights learnt with only 100 training cases. These weights are still far away from values where the formal analysis predicts that the deteriorating effect of the virtual attribute vanishes.

Since even increasing the number of training cases does not improve performance in both RELIEF and IB4, we think that the effect of the virtual attributes is too small to have an effect on the weight-learning method. This means that inaccurate virtual attributes that deteriorate accuracy will not be filtered away during the weight-learning process. This result is not as bad as it seems, because inaccurate virtual attributes only have a deteriorating effect in a small interval of inaccuracy.

6.5 Incorporating contextual knowledge into weight learning

Generally, weight-learning methods can benefit from domain knowledge. For example, all methods of incorporating domain knowledge into similarity measures can be used implicitly for weighting methods that have a performance bias, since they make use of the similarity measure. Thus, the knowledge in the similarity measure biases the weight-learning.

Furthermore, performance bias methods implicitly use the knowledge that there are several attributes. In contrast, most of the methods in section 6.3 with a preset bias learn the weight of attributes individually while being oblivious to the fact that there are other attributes. Thus, interferences and dependencies between attributes can only be handled by methods with a performance bias. But those methods use the knowledge implicitly, while we are more interested in ways to explicitly use domain knowledge. In this section we explicitly incorporate contextual knowledge into the weight-learning method RELIEF (Kira & Rendell, 1992).

In section 5.5.1 we applied contextual knowledge in order to express that an attribute was not relevant in a certain region. If we use traditional weight-learning to learn the weights of the remaining attributes, there are the following issues. First of all, global weight-learning methods such as RELIEF will converge on weights that try to express the relevance of attributes for the whole instance-space. This contradicts the bias of the knowledge-rich similarity measure which treats some attributes as irrelevant only in certain regions. Second, the weight-learning methods cannot exploit the knowledge that in certain regions some attribute weights should be exactly 0.

We tackle these issues by incorporating contextual knowledge into the weight-learning method RELIEF. The RELIEF algorithm works as follows: It randomly selects case x from the case-base, retrieves the most similar case p with the same class from the case-base, and the most similar case n with a different class. Then the weight w_i of each attribute A_i is updated by using the formula

$$w_i = w_i - \delta(A_i(x), A_i(p)) + \delta(A_i(x), A_i(n)),$$

where

$$\delta(A_i(c_1), A_i(c_2)) = \begin{cases} |A_i(c_1) - A_i(c_2)| & : \; iff\ A_i\ numerical \\ 1 & : \; iff\ A_i\ nominal, A_i(c_n) = A_i(c_m) \\ 0 & : \; iff\ A_i\ nominal, A_i(c_n) \neq A_i(c_m) \end{cases}$$

This procedure is repeated m times. In (Kononenko, 1994) it has been proposed to modify the algorithm to repeat the weight-update once for each training case.

Contextual knowledge of the form $relevant(A, c) \leftarrow \rho(c)$ can be incorporated as shown in the following PSEUDO-code:

```
set the default weight of all attributes to 0
repeat for each training case x
    retrieve most similar case p with the same class as x
*    (using only attributes satisfying relevant(A, x))
    retrieve most similar case n with a different class than x
*    (using only attributes satisfying relevant(A, x))
    for all attributes Ai
*    that satisfy relevant(Ai,x)
        wi = wi - delta(Ai(x),Ai(p))+delta(Ai(x),Ai(n))
```

The lines marked with a star are new as compared to the standard RELIEF algorithm. We use the values learnt by RELIEF as weights and not as threshold criterion for filtering features (cf. (Wettschereck et al., 1997)). The changes have the effect that the weight for an attribute is not adjusted in regions where it is known to be irrelevant. Thus, if the relevance of an

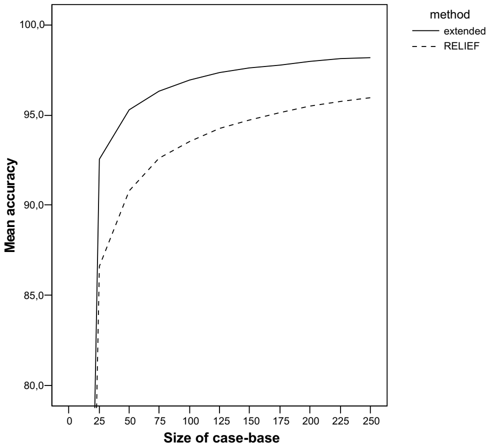

Figure 6.5: The accuracies obtained by the traditional RELIEF weight-learning algorithm and the knowledge-rich extension.

attribute is different in different regions, the global weight is not influenced by the regions where the relevance is 0.

In a three-dimensional instance space, the concept

$$f(c) \leftarrow (A_1(c) < 50 \vee 60 < A_1(c) < 90) \wedge 20 < A_2(c) < 80 \wedge 40 < A_3(c) < 60$$

has the shape of two cuboids that differ in their position and length on the A_1 dimension. We assume the contextual knowledge $relevant(A_2, c) \leftarrow A_1(c) > 50$. The standard similarity measure whose weights are learnt with the traditional RELIEF algorithm performs significantly worse than the similarity measure that uses the contextual knowledge and whose weights are learnt by the above weight-learning algorithm that exploits the contextual knowledge (see figure 6.5).

This means that contextual knowledge can also be used in a weight-learning method such as RELIEF. In this particular experiment, the contextual knowledge successfully separates the two concept parts and focusses the weight learning for A_1 on the region where it is relevant. In contrast, the traditional RELIEF algorithm has to converge on a weight that is a compromise between irrelevance (in the half of the instance-space with $A_1 < 50$) and high irrelevance (in the other half).

6.6 Conclusion

In this chapter we showed that learning the weights of virtual attributes and observables can drastically increase classification accuracy. Since the weights of observables have to be learnt anyway, the weights of virtual attributes can be learnt in the same process. We implemented several weight-learning methods with different biases and showed that the accuracy increase is not restricted to individual methods or particular biases. In the benchmark data set of the Promoter Gene Sequence domain, we achieved accuracies that exceed the best results reported in the literature so far. In the Japanese Credit Screening domain we did not find results in the literature, but the weight-learning methods provided accuracies that are better than the weighted knowledge-poor similarity measure.

Unfortunately, it seems that the weight-learning methods that we tested cannot be used to filter away those virtual attributes that are in the inaccuracy interval where they deteriorate classification accuracy. This finding supports our formal analysis which showed that binary virtual attributes are robust against weight changes. Only if the weight of a virtual attribute is very small, its effect on classification vanishes. The weight-learning methods did not converge to such low weights, although there is a tendency to give lower weights to worse virtual attributes.

Knowledge that is useful in similarity measures can also be used in weight-learning methods. We incorporated contextual knowledge into RELIEF and showed that the knowledge improves accuracy. Future work will have to investigate other weight-learning methods and knowledge types in order to test whether other incorporation methods are equally successful.

Chapter 7

Application to Multi-Agent Systems

> I came to the conclusion that the whole purpose of the simulation-league was to come up with coaching.
>
> *Gal A. Kaminka, Chair of the 2001 RoboCup simulation competition*

In this chapter we implement and evaluate our approach of enriching similarity measures with domain knowledge in the domain of opponent modelling in multi-agent systems. We show that the different knowledge types are useful in a complex domain such as simulated soccer.

7.1 Introduction

Opponent modeling is an essential part of performing well in adversary domains, as it allows to predict future actions of the opponent and adapt one's own policy accordingly. Case-based reasoning is a common method for opponent modeling in multi-agent systems (e. g. (Ahmadi et al., 2003; Wendler, 2004; Denzinger & Hamdan, 2004)), because it requires only few training instances and can describe any situation-action space which is important, as the different opponents may use various situation-action spaces for their decision-making. In multi-agent systems several autonomous agents are active. From a CBR perspective, the classification goal is to predict an agent A's action in a given situation c. The CBR system compares c to a case-base of previously observed situations, selects the situation c' that is most similar to c, and returns the action of A in c'.

MAS are a particularly interesting domain for our approach, because each case can be used for several classification goals. That is, for the same situation there are several actions taken by the agents. Since in our approach the classification goal is to predict an agent's actions, each agent's action can be seen as a classification goal. For example, the same situations can be used for classifying a defender's actions or a forward's actions. Different attribute weights and different attributes will be needed for the various classification goals (cf. (Ahmadi et al., 2003)).

Our assumption which serves as learning bias is that an agent behaves similarly in similar situations. This is certainly true for reactive agents which act mostly based on their perceptions of the world-state (cf. (Denzinger & Hamdan, 2004; Ahmadi et al., 2003) for a similar assumption). However, for deliberative agents which plan or even learn, this assumption may not hold. Still, in a highly dynamic domain such as RoboCup, planning can only be done for short time steps. Hence, we believe that our similarity assumption works for most agents. In any case, to validate this assumption, we made experiments with a variety of agents.

The classification- or prediction-accuracy of CBR depends on the quality of the similarity measure. Unfortunately, implementing a similarity measure for opponent modeling is not trivial due to a number of issues:

Context: Which situations should be regarded as similar depends on the context. The similarity measure must be adapted to the game situation and role of the agent whose action is to be predicted. Consider situations in a soccer game: The positions of team B's defenders will be rather irrelevant if the classification goal is the action of team A's goalie, but rather relevant if the classification goal is the action of team A's forwards. For these two classification goals, the similarity measure must weigh attributes (i. e. player positions) differently. Other CBR approaches in simulated soccer dealt with this problem by introducing a focus: Cases contain only positions of those players that are close to the ball (Ahmadi et al., 2003). Another method is to partition the known cases into defensive, transitional, and offensive sets (Marling, Tomko, Gillen, Alexander, & Chelberg, 2003). Yet, such methods are domain specific ad-hoc solutions.

Selecting features: The performance of similarity-based classification degrades with the number of irrelevant attributes (Griffiths & Bridge, 1996). However, even after successful filtering of irrelevant attributes, similarity measures can still be improved. Adding relevant abstract features can improve classification accuracy (Richter, 2003; Steffens, 2004c).

Sparseness of data: Usually, in CBR it is assumed that there is an abundance of data (Wilke & Bergmann, 1998). However, in opponent modeling this is not the case, since cases enter the case-base over time as observations are made. Thus, a requirement for similarity-based opponent modeling is that it has to perform well with sparse data. The less data is required, the sooner the system can provide good predictions. Fortunately, the lack of knowledge in the case knowledge container can be compensated by moving additional knowledge into the similarity measure knowledge container (Wess & Globig, 1994; Richter, 1995). In this chapter we explore how different forms of knowledge can be incorporated into similarity measures in order to make good predictions with sparse data.

Attribute matching: In most CBR applications, matching attributes is straight-forward, as equally named attributes or attributes at the same position of a vector are matched. However, when applying CBR to opponent modeling in multi-agent systems, matching of attributes is not trivial. The agents of the two situations have to be matched in a situation-specific way, so that their properties (e. g., their positions and velocities) can be compared. This matching has to take the agents' roles into account.

These issues make opponent modelling in multi-agent systems a challenge for case-based reasoning (and any other machine learning approach). We show how these issues can be tackled by incorporating domain knowledge into the similarity measure.

The next section describes the evaluation domain, simulated soccer. Section 7.3 defines which types of knowledge we used for similarity-based opponent modeling. In section 7.4 we motivate why we focus on high-level actions in our experiments. Afterwards, in section 7.5 we describe how some of the knowledge types can be represented using goal-dependency networks. The knowledge-poor and knowledge-rich similarity measures are defined in section 7.6. Section 7.7 will show how the knowledge-rich similarity measure can be used for attribute- as well as multi-agent matching. Section 7.8 reports the evaluation results of the implementations. Related work is discussed in section 7.9, and the last section concludes and outlines future work.

7.2 An Example Multi-Agent System: RoboCup

The RoboCup domain is a typical multi-agent system where opponent modeling is crucial for successfully counteracting adversary agents (Kitano et al., 1997; Ahmadi et al., 2003). Two teams of autonomous agents connect to a server and play simulated soccer against each other. Each player is an autonomous process. This is a challenge for opponent modeling, since the behavior of each opponent player has to be approximated.

Decision making is done in discrete time steps: Every 100ms the agents can execute a primitive action and the world-state changes based on the actions of all players and the game physics (Chen et al., 2001). Basically, the action primitives are *dash, turn, kick*, which must be combined in consecutive time steps in order to form high-level actions such as passes or marking. The agents act on incomplete and uncertain information: Their visual input consists of noisy information about objects in their limited field of vision. There is an additional privileged agent, the online coach, which receives noise-free and complete visual input of the playing field. The online coach has been introduced mostly for opponent modeling purposes. Every 100 ms it receives information about the position and velocity of all objects on the playing field (22 players and the ball). The agents' actions cannot be observed directly, but can be inferred from the differences between consecutive world-states. For instance, in our implementation the coach assumes that the player controlling the ball executed a kick, if the ball's velocity increases.

The match is always in one of several play-modes, such as ”free kick for left team”, ”ball out of bounds for right team”, or ”play on”. In the play-modes different rules apply. For example, in free kick-situations only one team is allowed to kick the ball.

Cases for our CBR system are generated from the observations of the coach. A case is represented in two parts: 47 attributes (23 positions, 23 velocities and the play-mode) specifying the situation, and 22 attributes storing the actions. In a query, only the situation is known and one of the actions serves as the classification goal; the other actions are ignored.

RoboCup is an ideal domain for evaluating our approach, because the same case-base can be used for different classification goals: The action of each player is handled as a single prediction task with its own classification goal. Since the coach receives information about all situation attributes, cases can be stored without further analysis. Generalization wrt. the specific classification goal is deferred until classification time, a property of lazy learning (Aha, 1997). Furthermore, the domain is complex enough so that we can use almost all of the knowledge types that we introduced before.

7.3 Types of Domain Knowledge

This section discusses which types of knowledge are useful for similarity-based opponent modeling. Previous similarity-based approaches used domain knowledge implicitly or in an ad-hoc kind of way (e. g. (Denzinger & Hamdan, 2004; Marling et al., 2003; Ahmadi et al., 2003)).

Multi-agent systems are situated in typically rich and complex domains (G. Weiss, 1999). Usually much domain knowledge exists, but problem-solving knowledge is missing. Remember that domain knowledge specifies which objects and relations between them exist in the domain (Bergmann et al., 1994), while problem-solving knowledge states which actions are to be taken given a state of the problem space and a goal. That is, in MAS it is known which objects and relations exist in the domain, but it is unknown how to develop a good policy or behavior for the agents. In our approach we use the fact that the behavior of the agents are constrained by the properties of the domain. For example, in simulated soccer, an agent should avoid being offside, otherwise his actions result in penalties. We exploit the explicit rules of soccer and the implicit structure of the domain by incorporating domain knowledge into the similarity measure.

In our evaluation we implemented the following types of knowledge (this is an extension of previous work (Steffens, 2005d)). For details about the implementation refer to section 7.6.

Transformational knowledge:

We used transformational knowledge in order to match situations from one wing of the field to the other. This was done by exploiting the information that the playing field can be mirrored. Using a mirroring operation, a situation where a forward attacks from the left and has a defender blocking his way to the goal can be matched to a situation where the forward attacks from the right with a defender blocking his way.

Inferential knowledge:

Based on the observable attributes we defined additional virtual attributes. The feature `pressing` is true if two or more players attack the ball-owner, that is, if two or more opponent players are within a radius of 5m around the ball.
Another virtual attribute is the `role` of the ball-owner. Roles are determined by the players' relative positions (see section 7.5). Note that `role` is a player-specific attribute, so that 22 additional attributes are introduced. However, we use only one weight for these 22 attributes and normalize them to blend in with the other attributes (such as `play_mode`) (see below). Before players (and their positions and velocities) are compared to each other, they have to be matched (see section 7.7)
`player_free` is another player-specific attribute that describes whether a player stands free. This is defined as having no opponent around the player in a circle with the radius 5m. The radius was found empirically. Additionally, we introduced a player-specific attribute `player_behind_ball` which is true if the player is between the ball and the opponent goal.

Contextual knowledge:

Contextual knowledge is useful for similarity-based opponent modelling. In simulated soccer, the attributes describing team A's defenders are irrelevant if team A's forward has the ball and is close to the opponent goal. But if team B's forward has the ball, the attributes describing team A's defenders are highly relevant. Thus, these attributes are contextual and are dependent on the context defined by which team has the ball and by the ball's position. We used a rule-base that determined in which contexts the attributes were relevant.

Matching knowledge

We treat all break play-modes (that is, all modes except "play on") as equivalent. These play-modes are special since play only resumes if the ball is passed by a player of the appropriate team.

At the moment, there is an effort to represent soccer knowledge as qualitative knowledge (Dylla et al., 2005). While the notion of "qualitative" is not strictly defined in the knowledge representation and reasoning community, in most work it means to discretize numerical parameters into a small set of intervals. For example, the soccer field is partitioned into regions such as penalty-area, left wing, right wing. This is equivalent to our notion of matching knowledge, where the exact value of an attribute is not important as long as it is in a certain interval. The same idea is used in (Dylla et al., 2005): Positions in the same region are assumed to be equivalent with respect to decision making.

We used the matching knowledge to partition the field into a set of regions (directly in front of goal, penalty area, opponent corners, and wings). Note that in our approach we also make use of transformational knowledge as we mirror the two opponent corners and the two wings.

Apart from the matching knowledge, maybe even more types of knowledge can be found in such a qualitative theory of soccer. However, the details are not yet published and not available to us.

7.4 Focus on high-level actions

In a complex domain such as RoboCup it is infeasible to predict an agent's behavior in terms of primitive actions. For individual skills (e. g. dribbling), primitive actions are often combined by neural networks. These are trained using amounts of training instances that are typically one or two levels of magnitude greater than the amount of observations available for opponent modeling. Hence, it is infeasible to predict an agent's primitive actions. Rather, in our experiments we predict the high-level action *shoot on goal*. We assume that for taking countermeasures it is sufficient to anticipate high-level actions within a certain time window. For example, if a defender knows that within the next 20 time steps an opponent will shoot on the goal, it can position itself accordingly (and maybe even inhibit the shot by doing so. Therefore, in our prediction experiments the agents do not use the predictive information in order not to interfere with the prediction accuracy.) For

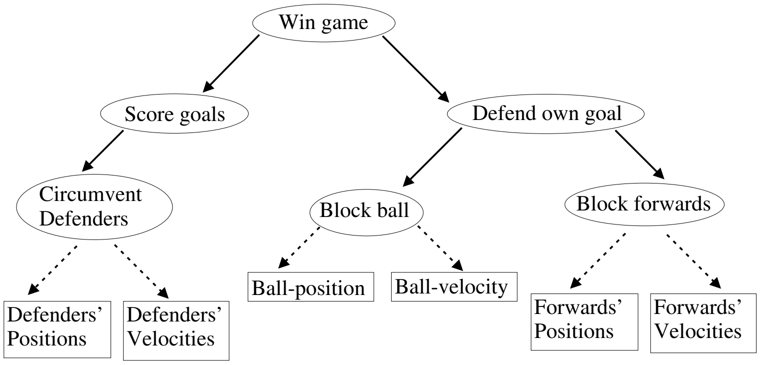

Figure 7.1: Vague domain knowledge represented by a goal dependency network (GDN). Goals and subgoals are depicted as ellipses, attributes as rectangles, virtual attributes as dotted rectangles. Dotted arrows connect goals to properties, solid lines connect subgoals to goals.

both the static and the adaptable similarity measures, the prediction of an action is counted as correct if the action occurs within 20 time steps after the prediction.

Note that recognizing high-level actions is not trivial, since the recognition has to aggregate several consecutive observations. We used recognition methods from our earlier works (Steffens, 2002), which provides acceptable recognition rates.

7.5 A first experiment: Goal-dependency networks

In a first experiment we tested whether inferential and contextual knowledge can really be useful in a complex domain such as RoboCup. This experiment is preparatory in nature, as it does not consider knowledge types such as transformational or matching knowledge. To represent the knowledge we used goal-dependency networks (GDNs) as proposed in (Stepp & Michalski, 1986). More details about this experiment can be found in (Steffens, 2004a).

7.5.1 Using the Goal-Dependency Network

In GDNs the agent's goals are divided into subgoals and linked with them. Attributes are connected to a subgoal if they are relevant to achieve that subgoal. For example, the goal `defend_own_goal` can be divided into the subgoals `block_forwards` and `block_ball`. A relevant attribute for `block_ball` is the `ball_position` (see Figure 7.1).

We can apply our framework in order to analyze the knowledge contained in GDNs. From the perspective of the knowledge type hierarchy, GDNs are a combination of inferential knowledge (about virtual attributes) and of contextual knowledge (a property is important if a certain subgoal is active).

A GDN can be exploited to select attributes based on the situation (cf. context-dependence in section 7.1). To do this, the part of the GDN-tree that is relevant for the situation must be determined. First, the estimated goal of the agent serves as an entry point into the GDN. The goal is estimated by an explicit mapping of the modeled agent's role onto a node in the GDN, e. g. $\{\langle forward, score\_goals\rangle, \langle generic, win\_game\rangle, \langle defender, defend\_own\_goal\rangle\}$. For example, for predicting the action of a forward, the subtree under *score_goals* will be activated.

The inferred role must not necessarily reflect the role that the programmer of the opponent agent intended. Rather, it must capture the situation-specific function of the agent in the particular situation. For our experiments, we approximated agent roles (forwards, midfielders, defenders, goalie) by their positions on the field. It might be the case that the team dynamically changes roles, that is, a player might switch from one role to another (cf. (Kuhlmann, Stone, & Lallinger, 2005)). Accordingly, our system does not infer roles statically, but reconsiders the role of each agent for every situation: During a manually selected observation window, the average x-coordinate (the x-axis runs from one goal to the other) of each player of a team is calculated. The players are then sorted according to their average x-coordinate. The player with the smallest coordinate is assumed to be the goalie. The rest of the sorted list is analyzed for the two largest gaps between consecutive players. These gaps are assumed to be the dividing line between forwards, midfielders, and defenders. Although this is a rough approximation of the true roles, the method proved to be accurate enough for our purposes (cf. (Steffens, 2004a)).

After identifying a node G as the entry point into the GDN, the relevant attributes are inferred via backward-chaining. The whole subtree of G is activated recursively, and the attributes of each activated subgoal-node are added to the set R, the so-called respects of similarity (Medin et al., 1993). When using GDNs, attributes can only be divided into relevant vs. irrelevant. The similarity measure is adapted by setting the weight of attributes in R to 1 (relevant), and all the other ones to 0 (irrelevant). That is, relevant attributes receive the maximum weight, and irrelevant attributes receive the minimal weight. Hence, the similarity measure for predicting actions of a defender differs from the similarity measure for predicting actions of a forward.

7.5.2 The Similarity Measures for Experiment 1

In this experiment, we use a standard knowledge-poor similarity measure as the baseline. As in the previous chapters, such a measure uses all available attributes and normalizes them into [0,1]. Thus, the static (i. e. knowledge-poor) similarity measure is defined as follows:

$$\begin{aligned} &sim(c_1, c_2) = \\ &\sum_{i=1}^{22} [\omega_i \cdot \Delta(p(i, c_1), p(i, c_2)) + \\ &\quad \omega_i' \cdot \Delta(v(i, c_1), v(i, c_2))] + \\ &\quad \omega_0 \cdot \Delta(bp(c_1), bp(c_2)) + \omega_0' \cdot \Delta(bv(c_1), bv(c_2)) + \\ &\quad \omega_{23} \cdot 1(pm(c_1), pm(c_2)) \end{aligned} \tag{7.1}$$

where c_1 and c_2 are the two situations in comparison, $p(i, c_j)$ and $v(i, c_j)$ are the position and velocity of player i in situation c_j, respectively, $bp(c_j)$ and $bv(c_j)$ are the ball-position and ball-velocity in c_j, respectively. $\Delta(A, B)$ is the Euclidean distance between A and B. $1(A, B)$ is 1 iff $A = B$, and 0 otherwise. $pm(c_i)$ denotes the playmode in a situation, and ω_k and ω_k' with $\sum_{k=0}^{23}(\omega_k + \omega_k') = 1$ are weights for positions and velocities, respectively. In this first experiment we set all weights to 1. In later experiments, weights were learnt using RELIEF.

The knowledge-rich similarity measure is adaptive to the situation for which the prediction is made. If attributes are deemed irrelevant in a situation, they are removed by setting their weight to 0. Relevance of the attributes is determined by using the GDN as described in section 7.5.1

In short, the knowledge-poor (static) similarity measure weights all attributes equally. The knowledge-rich (adaptable) similarity measure incorporates knowledge from the GDN depicted in Figure 7.1 and weights attributes

Table 7.1: Experiment 1: Mean accuracies of the static and the adapted similarity measures for various sizes of the case-base. p is the significance level of a paired, two-tailed t-test. N=48

Size of case-base	Static	Adaptable	p
1000	0.7499	0.7547	0.258
1500	0.7530	0.7605	0.08
2000	0.7558	0.7651	0.01
2500	0.7566	0.7685	0.003
3000	0.7573	0.7677	0.014
3500	0.7557	0.7665	0.013
4000	0.7670	0.7740	0.194
4500	0.7614	0.7734	0.057
5000	0.7719	0.7788	0.421

that are irrelevant for the classification goal with 0 and relevant attributes with 1 (cf. (Steffens, 2004a)).

7.5.3 Data

Both measures are tested on the same case-base and test cases. In this experiment we used 48 publicly available logfiles of recorded games [1] between 32 different teams. For each game, the two similarity measures were tested with various sizes of the case-base. A complete game lasts 6000 time steps, the case-base sizes ranged from 1000 to 5000 time steps with intervals of 500. The test cases were drawn from the remaining time steps at fixed intervals of 50 time steps. The classification goal was the action of the ball-owner. Hence, the role of the agent whose actions were to be predicted varied throughout the game.

7.5.4 Results

The mean accuracies of the static similarity measure and the similarity measure that was adapted to the classification goal are depicted in Figure 7.2. For the significance levels of paired two-tailed t-tests, refer to table 7.1.

[1] Available from http://www.carc.aist.go.jp/∼noda/RoboCup/LogFiles.

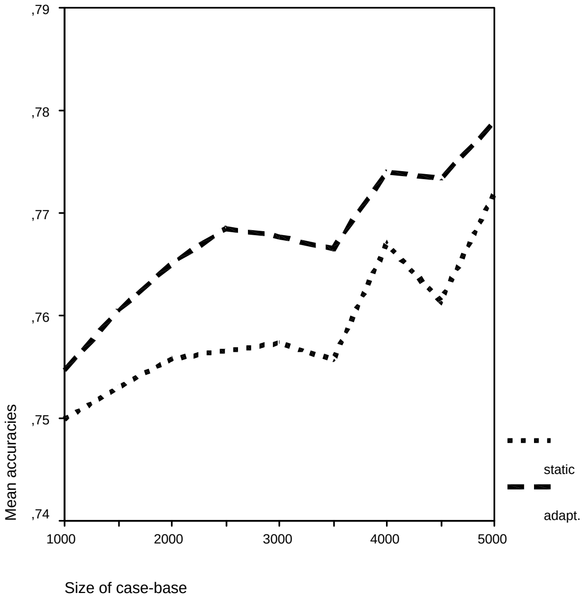

Figure 7.2: Experiment 1: Mean accuracies of the static and the adapted similarity measures for various sizes of the case-base. N=48

For all case-base sizes, the adaptable similarity measure achieved better accuracy values than the static measure. For most sizes, the difference is significant ($p < 0.1$, or even $p < 0.05$). This suggests that the prediction accuracy of CBR can be increased if the similarity-measure is extended by the knowledge contained in a GDN. More specifically, our results suggest that opponent modelling can benefit from taking into account the role or type of the agents. However, the difference between the two similarity measures is only small. We analyzed the retrieved cases and believe that the small impact is due to fixed formations in many teams: Players that were not directly involved in handling the ball stayed at fixed positions. E. g., whenever the forwards handled the ball close to the opponent goal, the defenders stayed at their home positions. Hence, the variance in positions is small, so that the static similarity measure retrieved the same cases as did the similarity measure that ignored defenders' positions for predicting the forward's action.
Still, the significant results show that using inferential and contextual knowledge yields better prediction results than the static similarity measure. Future research will consider non-binary weights in order to further increase accuracy.
General remarks about the experiment will also be discussed in section 7.8.6.

7.6 The Similarity Measures for Experiments 2 to 5

In the following experiments, we analyze different knowledge types.

7.6.1 Standard Measure

In all following experiments, we again use a standard knowledge-poor similarity measure. As in section 7.5.2 the standard measure is defined as follows:

$$\begin{aligned}
&sim(c_1, c_2) = \qquad (7.2)\\
&\sum_{i=1}^{22} [\omega_i \cdot \Delta(p(i,c_1), p(i,c_2)) + \\
&\quad \omega_i' \cdot \Delta(v(i,c_1), v(i,c_2))] + \\
&\quad \omega_0 \cdot \Delta(bp(c_1), bp(c_2)) + \omega_0' \cdot \Delta(bv(c_1), bv(c_2)) + \\
&\quad \omega_{23} \cdot 1(pm(c_1), pm(c_2)).
\end{aligned}$$

Any modification to this measure (such as removing or adding an attribute) would require additional domain knowledge. Note that even the standard similarity measure uses some domain knowledge, that is, it uses distributional knowledge, as it normalizes the ball and player positions and velocities.

7.6.2 Knowledge-Rich Measure for Experiment 2

The extended (knowledge-rich) similarity measure in experiment 2 is defined as follows:

$$\begin{aligned} \mathit{sim}'(c_1,c_2) = \omega_1 \cdot 1(\mathit{region}(c_1), \mathit{region}(c_2)) +{} \\ \omega_2 \cdot \left(\sum_{i=1}^{22} 1(\mathit{role}(c_1,i), \mathit{role}(c_2,i)) \right) /22 +{} \\ \omega_3 \cdot \beta(\mathit{playmode}(c_1), \mathit{playmode}(c_2)) +{} \\ \omega_4 \cdot 1(\mathit{pressing}(c_1), \mathit{pressing}(c_2)) +{} \\ \omega_5 \cdot \left(\sum_{i=1}^{22} 1(\mathit{free}(c_1,i), \mathit{free}(c_2,i)) \right) /22 +{} \\ \omega_6 \cdot \left(\sum_{i=1}^{22} 1(\mathit{behindBall}(c_1,i), \mathit{behindBall}(c_2,i)) \right) /22 +{} \\ \omega_7 \cdot \Delta'(\mathit{positions}(c_1), \mathit{positions}(c_2)) +{} \\ \omega_8 \cdot \Delta'(\mathit{velocities}(c_1), \mathit{velocities}(c_2)) \end{aligned} \tag{7.3}$$

where $1(X,Y) = 1$ iff $X = Y$, and 0 otherwise.
$region$ makes use of matching knowledge as it defines regions in which positions are treated as equivalent. $region(c) \in \{inFrontOfGoal, penaltyArea, corner, wing, midfield\}$ denotes the region the ball is in. Note that no distinction is made between left and right wing, and between the four corners. It should be noted that in this scenario, the definition of $region$ must be considered as possibly inaccurate, since it is not known whether the opponent uses the same definitions.
$playmode(c)$ determines the play-mode in situation c. There are about 20 different playmodes (Chen et al., 2001) and $\beta(Playmode1, Playmode2)$ is true iff $PlayMode1 = PlayMode2 \vee (PlayMode1 \neq \text{"}playOn\text{"} \wedge PlayMode2 \neq \text{"}playOn\text{"})$.

The positions and velocities of the 22 players are subject to contextual knowledge. That is, Δ' works as follows:

$$\Delta'(positions(c_1), positions(c_2)) = \frac{\sum_{i=1}^{22} g(pos(c_1, i), pos(c_2, m(i)))}{r},$$

where $pos(c, i)$ is the position of player i in situation c, $m(i)$ is the player in c_2 who corresponds to player i in c_1 (refer back to section 7.7), r is the number of agents that are relevant in situation c_1. $g(A, B)$ returns the similarity of the two positions A and B if the agent is relevant, and 0 if the agent is irrelevant. Relevance is determined based on the agent's role by the following rules:

- If the ball-owner is a forward, then the forwards from his own team, and the defenders and the goalie of the other team are relevant.
- If the ball-owner is a defender, then the defenders and midfielders of his own team, and the midfielders and forwards of the other team are relevant.
- In other situations, all players are relevant.

Of course also the contextual knowledge is possibly inaccurate, since the opponent may determine relevance of objects differently.

The predicates *pressing*, $free$, *behindBall* and *role* make use of inferential knowledge. $pressing(c)$ checks whether pressing is performed in the situation, that is, whether the opponent attacks the ball owner with two or more players (defined as a situation where two opponents are closer than 5m to the ball). $free(c, i)$ checks whether player i stands free, that is, no opponent player is within 5m distance of i. Since the parameters for $free$ and *pressing* are guessed, they are another example for possibly inaccurate domain knowledge. $behindBall(c, i)$ is true if the player i is between the ball and the opponent goal. $role(c, i) \in \{forward, defender, midfielder\}$ denotes the role of player i as described in section 7.5.

We applied the RELIEF method (Kira & Rendell, 1992) for learning the attribute weights. As noted before, we aggregate player-specific attributes such as $free$ and *role* so that only one weight is used for all of them, instead of 22. RELIEF does not have to be modified by this, because this is encapsulated in the delta-function (refer back to section 6.5), which calculates the difference of two cases for a given attribute:

$$delta(c_1, c_2) = \frac{\sum_{i=1}^{22} d(A(c_1, i), A(c_2, i))}{22},$$

where d is the local similarity of attribute A.

7.6.3 Knowledge-Rich Measure for Experiment 3

In experiment 3 we compare the standard measure with another knowledge-rich measure sim''. This latter measure uses only transformational knowledge:

$$
\begin{aligned}
& sim''(c_1, c_2) = max\Big\{ \qquad (7.4)\\
& \sum_{i=1}^{22} [\omega_i \cdot \Delta(p(i, c_1), p(i, c_2)) + \\
& \quad \omega'_i \cdot \Delta(v(i, c_1), v(i, c_2))] + \\
& \quad \omega_0 \cdot \Delta(bp(c_1), bp(c_2)) + \omega'_0 \cdot \Delta(bv(c_1), bv(c_2)) + \\
& \quad \omega_{23} \cdot 1(pm(c_1), pm(c_2)) \ , \\
& \\
& \sum_{i=1}^{22} [\omega_i \cdot \Delta(p(i, trans(c_1)), p(i, c_2)) + \\
& \quad \omega'_i \cdot \Delta(v(i, trans(c_1)), v(i, c_2))] + \\
& \quad \omega_0 \cdot \Delta(bp(trans(c_1)), bp(c_2)) + \omega'_0 \cdot \Delta(bv(trans(c_1)), bv(c_2)) + \\
& \quad \omega_{23} \cdot 1(pm(trans(c_1)), pm(c_2)) \Big\}
\end{aligned}
\tag{7.5}
$$

$trans(C)$ mirrors the situation's positions and velocities vertically, so that left and right wing are swapped. This means that transformational knowledge is used, as all positions and velocities of the cases in the case-base are compared to the query once in their original form and once vertically mirrored. The form which yields a higher similarity determines the similarity value of a query-case pair.

7.6.4 Knowledge-Rich Measure for Experiment 4

In experiment 4 we added inferential knowledge to the measure of experiment 3, so that the resulting measure used transformational knowledge and virtual attributes.

$$
\begin{aligned}
& sim'''(c_1, c_2) = max\Big\{ \\
& \quad \omega_1 \cdot \left(\sum_{i=1}^{22} 1(role(c_1, i), role(c_2, i)) \right) /22 + \\
& \quad \omega_2 \cdot 1(playmode(c_1), playmode(c_2)) + \\
& \quad \omega_3 \cdot 1(pressing(c_1), pressing(c_2)) + \\
& \quad \omega_4 \cdot \left(\sum_{i=1}^{22} 1(free(c_1, i), free(c_2, i)) \right) /22 + \\
& \quad \omega_5 \cdot \left(\sum_{i=1}^{22} 1(behindBall(c_1, i), behindBall(c_2, i)) \right) /22 + \\
& \quad \omega_6 \cdot \Delta(positions(c_1), positions(c_2)) + \\
& \quad \omega_7 \cdot \Delta(velocities(c_1), velocities(c_2)) \ , \\
& \\
& \quad \omega_1 \cdot \left(\sum_{i=1}^{22} 1(trans(role(c_1), i), role(c_2, i)) \right) /22 + \\
& \quad \omega_2 \cdot 1(playmode(trans(c_1)), playmode(c_2)) + \\
& \quad \omega_3 \cdot 1(pressing(trans(c_1)), pressing(c_2)) + \\
& \quad \omega_4 \cdot \left(\sum_{i=1}^{22} 1(free(trans(c_1), i), free(c_2, i)) \right) /22 + \\
& \quad \omega_5 \cdot \left(\sum_{i=1}^{22} 1(behindBall(trans(c_1), i), behindBall(c_2, i)) \right) /22 + \\
& \quad \omega_6 \cdot \Delta(positions(trans(c_1)), positions(c_2)) + \\
& \quad \omega_7 \cdot \Delta(velocities(trans(c_1)), velocities(c_2))\Big\} \qquad (7.6)
\end{aligned}
$$

Note that no matching knowledge is used: *region* has been removed, and playmodes are tested for identity only. Also no contextual knowledge is used (positions and velocities are compared using the Euclidean distance only, as specified by Δ).

7.6.5 Knowledge-Rich Measure for Experiment 5

In experiment 5 we slightly modified the measure from experiment 2 by removing the matching knowledge in the form of *region*.

$$
\begin{aligned}
sim''''(c_1, c_2) = \omega_1 \cdot \left(\sum_{i=1}^{22} 1(role(c_1, i), role(c_2, i)) \right) /22 \, + \\
\omega_2 \cdot \beta(playmode(c_1), playmode(c_2)) \, + \\
\omega_3 \cdot 1(pressing(c_1), pressing(c_2)) \, + \\
\omega_4 \cdot \left(\sum_{i=1}^{22} 1(free(c_1, i), free(c_2, i)) \right) /22 \, + \\
\omega_5 \cdot \left(\sum_{i=1}^{22} 1(behindBall(c_1, i), behindBall(c_2, i)) \right) /22 \, + \\
\omega_6 \cdot \Delta'(positions(c_1), positions(c_2)) \, + \\
\omega_7 \cdot \Delta'(velocities(c_1), velocities(c_2)). \qquad (7.7)
\end{aligned}
$$

Considering the large number of logfiles that have to be used for each experiment, it is infeasible to test all combinations of knowledge types. However, with the measures that we defined (and with the results of the GDN experiments), the experiments will provide enough data to form some hypotheses about useful knowledge types for similarity-based opponent modelling.
Before we look at the results of the experiments, we have to deal with the problem of attribute matching, which we describe in the next section.

7.7 Multi-Agent Matching

In most CBR applications, matching attributes is straight-forward, as equally named attributes or attributes at the same position of a vector are matched. However, when applying CBR to opponent modeling in multi-agent systems, matching of attributes is not trivial. For example, the positions and velocities stored in the cases are linked to specific players. Since it is rather common to swap positions in RoboCup (e. g. to move a tired player from an exhausting to a slower position), comparing positions of situation S_1 to situation S_2 must take into account that it is not necessarily the case that the position of player number 3 in situation S_1 must be compared to the position of player number 3 in situation S_2. Instead, it might be the case that in S_2 players 3 and 9 swapped positions. In that case, the desired concept of similarity can only be achieved by comparing the positions of players 3 and 9.

Hence, before two situations can be compared the agents of the two situations have to be matched. Traditional multi-agent matching usually requires a 1 to 1 matching (Stolzenburg, Murray, & Sturm, 2003). However, when doing multi-agent matching for attribute matching, this requirement must be lifted. Consider the example soccer situation in Figure 7.3 (top). The players A,B,C belong to one team and x,y,z to the other team. The situation on the right differs from the one on the left only in that player x has been moved. An optimal 1 to 1 matching (minimizing the summed distances) would match each player from the left situation to itself in the right situation. However, the relevance of player x is different in both situations. On the left, player x marks player C so that A cannot safely pass the ball to C. On the right, player x is basically equivalent to player z and does not mark player C. Hence, we propose to match both players x and z on the right to player z on the left, and to leave player x on the left unmatched, as no player on the right corresponds to its situation-specific role of marking.

Therefore, in our implementation multi-agent matching is done by matching those players of the same team that are most similar with respect to the similarity-measure. For each pair of player the similarity is calculated. This value is calculated by the normal similarity-measure, including spatial closeness and all player-specific virtual attributes, such as mirroring and knowledge of the role of the player. Each player is then matched with the player who achieves the highest similarity. In other words, the matching is done in a way as to maximize the similarity measure. We allow N to 1 matchings, because several players can have the same situation-specific role as another player in the other situation.

The knowledge-rich virtual attributes are also used for multi-agent matching. This is different than previous work in multi-agent matching in RoboCup where players were matched based on their spatial distance only (Stolzenburg et al., 2003).

To illustrate why virtual attributes are also useful for multi-agent matching, consider the example soccer situation in Figure 7.3 (bottom). The left bottom situation is the same one as the left top one. It differs from the right bottom one only in the position of player x. A matching algorithm that computes player similarity only based on spatial distance would assign player x and y from situation 4 to player y in situation 3. However, in situation 4 player

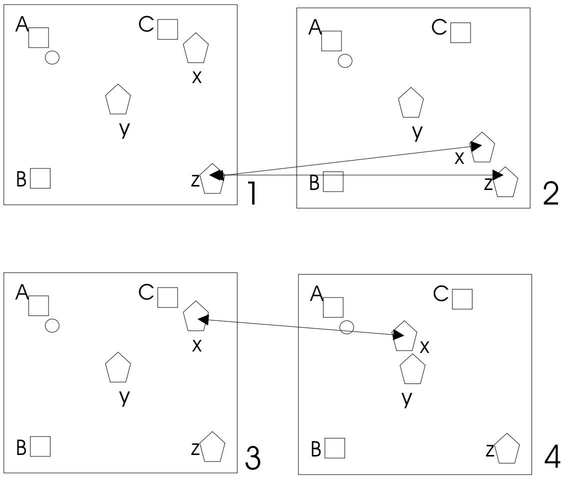

Figure 7.3: Player matching in soccer situations. Players A,B,C belong to one team, player x,y,z to the other. Unless otherwise depicted by arrows, players from the left are matched to themselves on the right. Top: Situation 1 and 2 are compared. Bottom: Situation 3 and 4 are compared.

x is not equivalent to player y since it may intercept a pass from player A to player C, just as player x in situation 3. Thus, a virtual attribute $betweenBallAndPlayer(X, .)$ is useful which is true for player x in both situations. If it is weighted great enough so that it outweighs the spatial distances, player x from situation 1 will be matched to player x in situation 2, which is consistent with the player's situation-specific roles.

Additionally, in our matching algorithm, the ball-owners of two situations are always matched, and of course players are only matched to players of their own team. The former fact is reminiscent of work in analogy (Markman & Gentner, 1990) where for example people in a situation are matched to each other based on the function or role they fulfill.

Since some contextual attributes in the similarity measure specify that some players are irrelevant (for example, team A's defenders are deemed irrelevant if team A's forward has the ball) in certain situations, the matching algorithm does not match these players in the corresponding situations.

7.8 Experiments

The following experiments tested whether the prediction accuracy for player actions increases if the similarity measure is extended with imperfect domain knowledge and which types of domain knowledge are particularly effective for this prediction task. That is, the standard measure was compared to the knowledge-rich measures. Both the unextended and the extended similarity measures are tested on the same case-base and test cases. The test domain is RoboCup.

7.8.1 Data

For the experiments 2 to 5, we used 51 publicly available logfiles[2] of recorded games between 21 different teams. For each game, the first W cycles of the match were recorded into the case-base. A complete game lasts 6000 time steps. The test cases were drawn from the remaining time steps at fixed intervals of 50 time steps. The maximal value for W was 4000 in order to have enough test cases remaining. The classification goal was the action of the ball owner.

The attribute weights were learned with RELIEF for all similarity measures (including the knowledge-poor measure). For each game the weights were relearned using the case-base as training data[3].

7.8.2 Results Experiment 2

In experiment 2 the standard measure was compared to sim', which used all knowledge types that were available. The mean prediction accuracies of both similarity measures are shown in figure 7.4.

For small case-bases that contain less situations than half the game, the prediction accuracy of the knowledge-rich measure is greater than the accuracy of the knowledge-poor one. However, if more data is available, the standard measure outperforms the extended measure. In statistical terms, for case-base sizes smaller than 2500, the extended measure is different from the standard measure with probability greater than 99.95% in a two-tailed t-test. For case-base sizes greater than 3250, again the extended measure is

[2] Available from http://www.carc.aist.go.jp/∼noda/RoboCup/LogFiles.

[3] Later analysis revealed that the ordering of the weights remained constant over games (and therefore constant over teams, too).

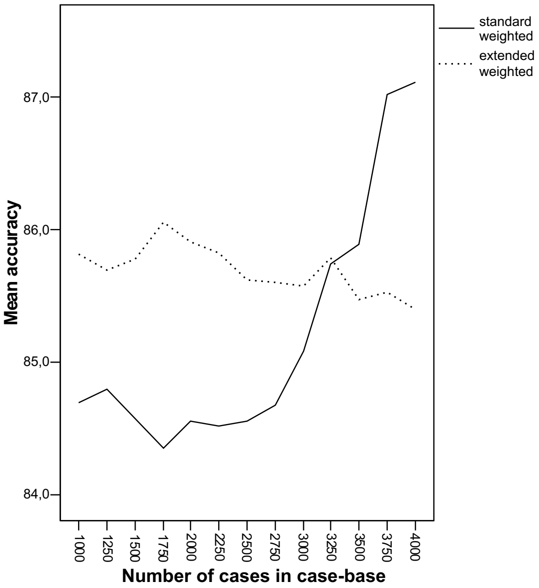

Figure 7.4: Experiment 2: Mean accuracies of the standard and the extended similarity measure sim' for various sizes of the case-base.

different from the standard measure with probability greater than 99.95% in a two-tailed t-test.

The dominance of the knowledge-rich measure for small case-bases is no surprise, since the knowledge container approach states that lack of case knowledge can be accommodated by additional knowledge in the similarity measure. But the fact that for larger case-bases this dominance is reversed, is surprising. This reversal is due to the fact that the accuracy of the knowledge-rich measure remains rather constant across the different case-base sizes. It even has a tendency to become smaller for greater case-base sizes. The accuracy of the standard measure behaves as expected and increases with the case-base size.

We believe that the accuracy of the knowledge-rich measure remains constant because with the incorporation of imperfect knowledge it has already reached an asymptotic level so that additional case knowledge does not add more information. Since the knowledge is imperfect, the asymptotic level is lower than the asymptotic level of the standard measure at greater case-base sizes. Even more so, as additional cases enter the case-base, they can conflict with

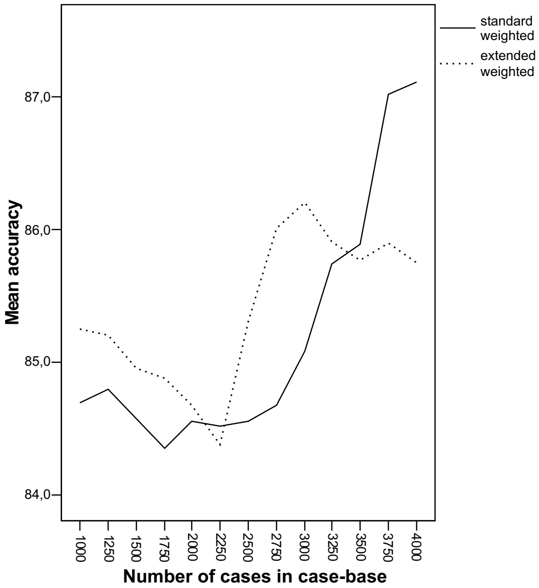

Figure 7.5: Experiment 3: Mean accuracies of the standard measure and *sim*″ (transformational knowledge only) for various sizes of the case-base.

the imperfect domain knowledge of *sim*′ so that the accuracy decreases.

Regarding the challenges of this domain, opponent modeling usually does not have plenty of data, that is, the system can not wait until late in the game. In particular, a modelling approach that has to accumulate observations for the whole first half of the game is useless. Thus, a method that performs well with few and medium amounts of data should be preferred. The measure *sim*′ satisfies this condition.

As a first result, we can conclude that our approach of incorporating imperfect domain knowledge into the similarity measure increases the accuracy of similarity-based opponent modelling. In the next experiments we investigate which types of knowledge contribute mostly to this good performance.

7.8.3 Results Experiment 3

In experiment 3 the standard measure was compared to *sim*″ which uses only transformational knowledge. The mean prediction accuracies of both similarity measures are shown in figure 7.5.

Remember that the curve of the standard measure is the same as in the previous experiment.

The most striking difference to experiment 2 is that the knowledge-rich measure is not constant across the case-base sizes, but has a valley-shaped form for small case-base sizes and a peak-sized form for greater case-base sizes. This behavior is due to the domain-specific characteristics of soccer. In the early parts of a game, it is unlikely that a given situation occurs once on both wings. Now assume a situation on the left wing is used as query. By the transformation of vertically mirroring, this situation is reflected to the right wing. But it is very unlikely that in the early parts this situation has already occurred on the right wing. Instead, due to the mirroring the query might now have become more similar to situations on the right wing that occurred e. g. closer to the goal. Thus, the horizontal difference is counter-weighted by the reduced vertical difference after mirroring. Actions will be different depending on the horizontal position so that the retrieved case provides the wrong prediction.

For a certain time, the probability of retrieving bad cases will increase as the case-base gets fuller. However, there is a threshold where the case-base gets so full, that it becomes more likely that situations occur on both wings. From that time on, the prediction accuracy will increase again.

The results suggest that transformational knowledge should only be used with medium case-base sizes.

7.8.4 Results Experiment 4

In experiment 4 the standard measure was compared to sim''' which uses virtual attributes and transformational knowledge. The mean prediction accuracies of both similarity measures are shown in figure 7.6.

The accuracy curves shows that adding virtual attributes to sim'' (which used only transformational knowledge) makes the valley disappear. Furthermore, the accuracy of sim''' has the intuitive behavior to increase with the case-base size. Unfortunately, there is no significant difference to the standard measure.

7.8.5 Results Experiment 5

In experiment 5 the standard measure was compared to sim'''' which is a variant of sim'' and uses no matching knowledge. The mean prediction accuracies

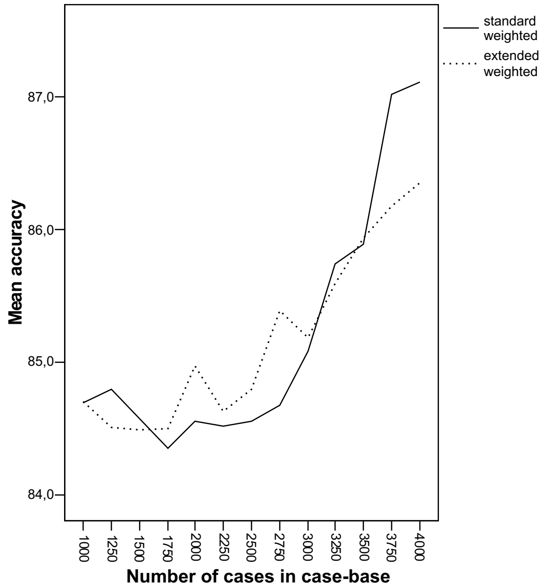

Figure 7.6: Experiment 4: Mean accuracies of the standard measure and sim''' (virtual attributes and transformational knowledge) for different CB sizes.

of both similarity measures are shown in figure 7.7.

Apparently, the accuracy curve of sim'''' is not significantly different from the curve of sim'' (it might be slightly lower, but not significantly so with this number of games). This suggests that the matching knowledge contained in the definition of regions on the field was either too different from the definitions used in the teams, or does not have a major impact on the prediction accuracy in general. Since the implementation of the teams is not public, it is impossible to decide which of these two explanations are correct. However, this should be addressed in future work, since the recent efforts to describe qualitative soccer knowledge (which makes use of matching knowledge by discretizing positions and directions) will possibly introduce matching knowledge into the decision processes of several teams.

This result is also consistent with our analysis of the learnt attribute weights. The weight of the *region* attribute was always very low which suggests that its relevance for the prediction task was also low. In fact we conducted this particular experiment only because the weight was so low.

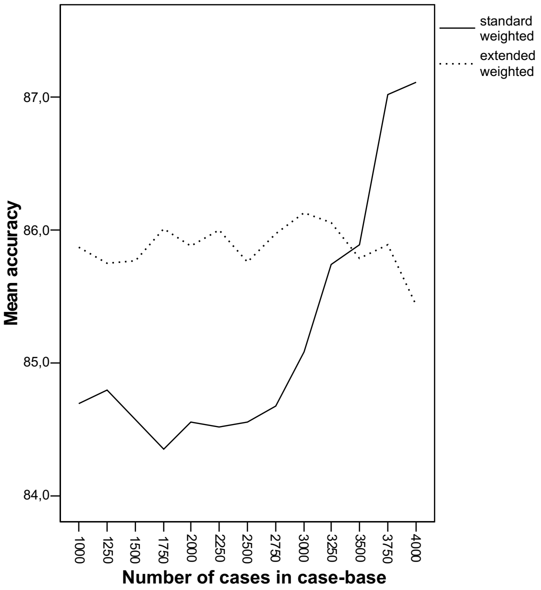

Figure 7.7: Experiment 5: Mean accuracies of the standard measure and sim'''' (without matching knowledge) for various sizes of the case-base.

7.8.6 Results: General Discussion

The accuracy difference between the extended and non-extended measures is always small. Our analysis suggests that the small impact of the extended similarity measures is due to the fact that any increase of prediction accuracy is difficult, since the behaviors of the player agents are implemented by many different methods. Player implementations range from simple decision trees (Buttinger et al., 2001), through probabilistic approaches (Boer, Kok, & Groen, 2002) to neural networks (Riedmiller, Merke, Nowak, Nickschas, & Withopf, 2003). Particularly if behaviors are learned, the partitioning of the situation space can be highly irregular and complex.

Furthermore, it is very unlikely that the opponent players used the same domain knowledge. Hence, their situation space will be different from the situation space of our case-base.

An additional factor in the experiments is the underlying noise. First of all, our system has available perfect information about the world-state, but the players that are to be predicted have noisy and incomplete information. Thus, an inherent problem of such an opponent-modelling approach is that it does not use the same information that the modelled agents possess. A second effect of the noisy domain is that actions cannot be recognized unambiguously (Steffens, 2002) and may come out differently than intended. Thus, if the coach correctly predicts that an agent will decide to execute a pass, the agent may fail to kick the ball, so that in the evaluation the prediction will be counted as wrong.

Note that no team-specific knowledge is used, but only general knowledge about the domain.

On the positive side, the knowledge-rich similarity measures sim'' and sim'''' are significantly better for small case-base sizes than the standard measure. This means that additional domain knowledge helps to predict the actions of agents even if it is imperfect. We cannot investigate the level of inaccuracy and incompleteness more closely since the target concepts are hidden in the implementation of the teams. Thus we can only state that the knowledge that we used was certainly incomplete and inaccurate, but still helped to increase the prediction accuracy.

Note that in our experiments the accuracies do not converge to some limits asymptotically. This is due to the fact that the amount of data available from a game of 6000 time steps is not sufficient to reach the limit.

7.9 Related Work

In this section we discuss alternative approaches to opponent modeling beyond case-based reasoning.

7.9.1 Assuming optimal opponents

The most simple form of opponent modelling is to assume that the opponent behaves optimally. For example, if a player wants to intercept the ball before the opponent does, it may assume that the opponent has a correct world model and uses the fastest path to the ball. This way, the agent can reason about the needed speed and power that it has to spend in order to get to the ball first. Or, if the needed power is not available, it can give up trying. (Stone, Riley, & Veloso, 2000) report that such opponent modelling can improve performance.

Such an approach is similar to the classic minimax-algorithms (Shannon, 1950) for search-trees in turn-based games and does not use specific knowledge of the opponent at all. It has been shown that as soon as the opponent is not optimal, overestimating its performance leads to wrong predictions and thus many chances are ignored (Jansen, 1990). Thus, in order to exploit flaws of the opponent, opponent-specific knowledge has to be acquired.

7.9.2 Similarity-based prediction

One method to represent opponent-specific knowledge is a similarity-based approach as proposed in (Denzinger & Hamdan, 2004), where pre-defined stereotypes of situation-action pairs and the nearest-neighbor-rule are used. The method was designed to generalize well with sparse observations and to be efficient when there are large amounts of observations. Just as in our approach, it is assumed that the modelled agent is reactive. However, the similarity measure is not systematically enriched with domain knowledge.

Opponent models used for similarity-based prediction are easy to acquire (Wendler, 2004). Observations of the opponent are stored as state-action pairs into a case-base. To reduce the required number of observations, (Ahmadi et al., 2003) proposes to use a second layer in the case-based reasoning system. This additional layer is used to retrieve parameters for the similarity measure and to decide where the focus is in the field. Cases are not stored for specific teams as in (Steffens, 2005d), but cases of different teams are stored together in one case-base. Outside of the domain of RoboCup, it has been proposed to use models of other agents as stereotypes for agents for which not enough observations are available (Denzinger & Hamdan, 2004).

Similarity-based approaches are easy to implement and maintain, but have the problem that expensive computations are deferred to the time-critical phase of online classification.

7.9.3 Model selection

One method to represent opponent-specific knowledge is to aggregate knowledge into a set of predefined opponent models. Each model in the set is assumed to roughly describe the behavior of a subset of all possible opponents. During play, the observations of the opponent are used to select the most probable model which is then used to generate predictions (e. g. (Riley & Veloso, 2002)) or to select a counter-strategy (e. g. (Visser, Drücker, Hübner, Schmidt, & Weland, 2001; Steffens, 2002)).

There are different types of opponent models. In (Visser et al., 2001), models describe typical formations. The observed positions of the opponent are fed into a trained neural network, which tries to classify them into a set of predefined formations. If a classification can be done, the appropriate counter-formation is looked up and communicated to the players.

In contrast, an opponent model is a set of state-action pairs in (Steffens, 2002). In this work, not the whole behavior is described in the manually created models, but only typical and salient tactical moves. A matching module then tries to match observed behaviors to the opponent models. Each model is associated with a counter-strategy. If a matching succeeds, the counter-strategy is sent to the players.

Another way to use opponent models is to predict future world states. Riley (Riley & Veloso, 2002) uses models that describe probability distributions of player positions. Observed player positions are classified by a Naive Bayesian classifier into a model. This model is then used to predict probable opponent positions given that the team moves the ball according to a given plan.

An inherent issue of model selection techniques is that they require a set of opponent models (which often have to be generated manually), and often require a mapping from models to counter-measures (which are typically handcrafted, too (Visser et al., 2001; Steffens, 2002)). In contrast, CBR only requires a set of observations, tuning of the similarity measure is optional. However, CBR as used in this paper does not provide an appropriate counter-action. Up to now it only predicts the opponent's actions.

7.9.4 Markov Decision Processes

Markov Decision Processes are well-suited to learn models that describe a sequence of observations. In (Riley, 2005), first Markov Chains are learnt from the observational data, which describe the transition probabilities between world-states. In order to reduce complexity and to better conform with the granularity of the coach language, abstract world states and actions are used. In order to transform the Markov Chains into a Markov Decision Process, the possible actions have to be provided, so that the transition probabilities can be expressed based on the action that the agents execute.

7.9.5 Rule learning

In order to learn and describe regular and systematic behaviors, rule-learning has been applied. (Ledezma, Aler, Sanchís, & Borrajo, 2005) uses C4.5 to learn decision trees for predicting the action type (such as dash, turn, kick). Then M5 is used to learn regression trees in order to predict the action's parameter values (such as the dash power). While learning the action type was successful, the parameter estimation performed badly.

Passing-behavior has been learnt in (Riley, 2005). First the positions of the passer, the pass receiver and all player positions were clustered. Then decision-tree learning was performed on the clusters of the passer locations, the clusters of receiver locations, and the angle and distance of all players to the ball. The learnt passing rules have been successfully used to mimic strong teams or to predict the behavior of opponents.

In (Kuhlmann et al., 2005) the opponent behavior is partitioned into offensive, defensive and formational modules, and each module is handled by a learning process. In experiments, it turned out that the formational learning was most successful, and that defensive and offensive learning did not contribute advantages.

While the above approaches learn rules from logfiles, in (Visser & Weland, 2003) propositional rules are learnt fast online for describing passing behavior and the goalkeeper.

7.9.6 Other approaches

In game theory there are approaches to learn opponent models from action sequences (Carmel & Markovich, 1996). Usually a payoff-matrix is necessary, but for predicting the opponent's actions this requirement does not hold (Rogowski, 2004). Unfortunately, these learning techniques assume that the opponent strategy can be described by a deterministic finite automaton, which might not always be the case in a complex domain. Most importantly, game theory can describe game states only as history of actions, which is infeasible in complex games such as RoboCup, where subsequent game states are not only determined by player actions but also by the game physics.

Predicting opponent actions can also be done via plan-recognition (Kautz, 1991). Predefined plan libraries are needed. Although recently there have been proposed approaches that can even handle reactive agents (Kaminka & Avrahami, 2004), to our knowledge plan recognition has not been applied to

RoboCup.

7.10 Conclusions and Outlook

We enriched similarity-based opponent modeling in multi-agent systems with imperfect domain knowledge. We showed how the knowledge types can be implemented and used in a complex domain such as RoboCup. This way we demonstrated that the knowledge types are not only present in artificial domains but can also be applied in simulations of the real world.
The prediction accuracies of knowledge-rich measures were compared to knowledge-poor measures in the domain of simulated soccer. The results suggest that similarity-based opponent modeling can benefit from domain knowledge even if it is imperfect and not known whether the opponent uses the same domain knowledge.
It is apparent that the accuracy curves did not reach an asymptotic level in our experiments. Thus, in future work it would be interesting to use several logfiles as case-base before a match in order to check when the approach reaches its maximum. If several logfiles are used, it will become necessary to use indexing approaches like case retrieval nets (Burkhard, 1998) in order to facilitate online processing and reduce the computational load.

Chapter 8

Conclusion

In this chapter we summarize the contributions of this thesis and outline directions for future work.

8.1 Contributions

We investigated how domain knowledge can be incorporated into similarity measures for classification. Specifically, we showed how different types of knowledge can be used to improve the accuracy of CBR with attribute-value representations.

We created a taxonomy of knowledge types that were previously researched in isolation. These types were proposed in psychology or in CBR focussing on structured representations. The contribution of this thesis is to formalize these types of knowledge, so that their differences and commonalities can be compared. As it turned out, several of these types stand in a type-subtype relation. That means, incorporation methods for a father type are also applicable to a child type. This systematization was only possible because we introduced new incorporation methods for several knowledge types, and also showed how existing incorporation methods can be applied to new knowledge types.

Furthermore, we investigated the effects of imperfectness of the domain knowledge. For empirical evaluation, the incorporation methods have been implemented and tested in several domains. These domains were artificial or from real-world sources. Some of these evaluations have been backed up with formal analysis. The empirical and formal investigations suggest that partial knowledge adds up, positively or negatively, depending on the inaccuracy of the knowledge. A surprising result was the influence of inaccurate virtual attributes on classification accuracy. It turned out that there is a certain interval of attribute inaccuracy in which the classification accuracy decreases. If the attribute inaccuracy is below this interval, classification accuracy increases beyond the baseline, and if the attribute inaccuracy is above the interval, the classification accuracy is equivalent to the baseline of a standard similarity measure.

For inconsistent virtual attributes the investigations suggest that the effect of a good virtual attribute is not deteriorated by another contradicting virtual attribute.

We believe that these findings will facilitate knowledge engineering for CBR systems, as the requirements of completeness and accuracy regarding the domain knowledge are softened. Of course, further research in real-world domains and applications will have to validate this hope.

Concerning the area of Machine Learning, our approach led to very good results in the Promoter Gene Sequences domain, where the accuracy in a leave-one-out evaluation exceeded the best known results from the literature. This suggests that CBR together with imperfect domain knowledge and weight-learning is a method that is as good as other learning mechanisms.

Finally, we evaluated our approach as knowledge-rich similarity-based opponent modelling in the complex domain of simulated soccer. The domain is rich enough so that the knowledge types can be implemented and tested. The results showed that incorporating general domain knowledge into the similarity measure can increase the prediction accuracy for agents.

8.2 Future work

Our analysis can be extended in several ways. First of all, since we focussed on attribute-value representations, it would be interesting to do a similar analysis for structured representations. We hypothesize that in principle the same knowledge types can be used for structured representations. Thus, a straightforward extension would be to investigate the effect of imperfect knowledge on similarity measures for object-oriented or graph-like representations.

Another interesting direction is to merge our results with approaches that actively learn domain knowledge by processing the cases in the case-base (e. g. (Stahl, 2004)). The requirements of domain knowledge (in terms of accuracy, consistency and completeness) can serve as goals or bias for learning mechanisms. As a special branch of such learning mechanisms, it seems promising to reactivate the research area of constructive induction, i. e. feature generation. As we have mentioned earlier, it sometimes appears necessary to modify intermediates in a domain theory, so that the feature is less specific or does not use vague sub-conditions anymore. Also, picking out sub-formulas that appear in several other intermediate concept descriptions should be a fruitful task for future work.

More work is needed in order to transfer our results from the classification domain to other domains that use similarity measures. In particular, clustering could benefit from knowledge-rich similarity measures so that meaningful clusters can emerge.

Finally, it would be most interesting to check whether and how the knowledge types that we proposed are valid for human similarity assessment, too. Our definition of knowledge types provides a clear terminology that can be used in cognitive science in order to set up new experiments about the effect of knowledge on similarity judgements. Such an analysis would complete the circle of psychologically motivated AI.

List of Figures

List of Tables

References

Aamodt, A. (1990). Knowledge-intensive Case-Based Reasoning and sustained learning. In L. Aiello (Ed.), *Proceedings of the 9th European Conference on Artificial Intelligence* (p. 1-6). London: Pitman Publishing.

Aamodt, A. (1994). Explanation-driven Case-Based Reasoning. In S. Wess, K.-D. Althoff, & M. M. Richter (Eds.), *Topics in Case-Based Reasoning* (p. 274-288). Springer.

Aamodt, A. (2001). Modeling the knowledge contents of CBR systems. In A. Aamodt, D. Patterson, & B. Smyth (Eds.), *Proceedings of the Workshop program at the Fourth International Conference on Case-Based Reasoning* (pp. 32–37).

Aamodt, A., & Plaza, E. (1994). Case-based reasoning : Foundational issues, methodological variations, and system approaches. *AI Communications*, *7*(1).

Aha, D. W. (1991a). Case-based learning algorithms. In R. Bareiss, S. Lewis, & I. Gravitis (Eds.), *Proceedings of the Case-Based Reasoning Workshop* (pp. 147–158). San Mateo: Morgan Kaufmann.

Aha, D. W. (1991b). Incremental constructive induction: An instance-based approach. In L. Birnbaum & G. Collins (Eds.), *Proceedings of the Eighth International Workshop on Machine Learning* (p. 117-121). Evanston, IL: Morgan Kaufmann.

Aha, D. W. (1992). Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms. *International Journal of Man-Machine Studies*, *36*(2), 267–287.

Aha, D. W. (1997). Editorial for the special issue: lazy learning. *Artificial Intelligence Review*, *11*, 7–10.

Aha, D. W., & Goldstone, R. L. (1992). Concept learning and flexible weighting. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society* (pp. 534–539). Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Ahmadi, M., Keighobadi-Lamjiri, A., Nevisi, M. M., Habibi, J., & Badie, K. (2003). Using a two-layered Case-Based Reasoning for prediction in soccer coach. In H. R. Arabnia & E. B. Kozerenko (Eds.), *Proceedings of the International Conference of Machine Learning; models, technologies and applications (MLMTA'03)* (pp. 181–185). CSREA Press.

Ahn, W. kyoung, & Dennis, M. J. (2001). Dissociation between categorization and similarity judgement: differential effect of causal status on feature weights. In U. Hahn & M. Ramscar (Eds.), *Similarity and categorization* (pp. 87–107). Oxford: Oxford University Press.

Ahn, W. kyoung, Kim, N. S., Lassaline, M. E., & Dennis, M. J. (2000). Causal status as a determinant of feature centrality. *Cognitive Psychology*, *41*, 361–416.

Althoff, K.-D., & Aamodt, A. (1996). Relating case-based problem solving and learning methods to task and domain characteristics: Towards an analytic framework. *AI Communications*, *9*(3), 109-116.

Anderson, J. R. (1983). *The architecture of cognition.* Cambridge, MA: Harvard University Press.

Armano, G., Cherchi, G., & Vargiu, E. (2004). Automatic generation of macro-operators from static domain analysis. In R. L. de Mantaras & L. Saitta (Eds.), *Proceedings of ECAI 2004* (pp. 955–956). IOS Press.

Atkeson, C. G., Moore, A. W., & Schaal, S. (1997). Locally weighted learning. *Artificial Intelligence Review*, *11*, 11-73.

Bain, A. (1855). *Senses of intellect.* London: J. W. Parker.

Barsalou, L. W. (1989). Intraconcept similarity and its implications for interconcept similarity. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning* (pp. 76–121). Cambridge: Cambridge University Press.

Baumeister, J., Atzmueller, M., & Puppe, F. (2002). Inductive learning for case-based diagnosis with multiple faults. In *ECCBR02 - Advances in Case-Based Reasoning* (Vol. 2416, p. 28-42). Berlin: Springer-Verlag. (Proceedings of the 6th European Conference on Case-Based Reasoning (ECCBR-2002))

Bergadano, F., & Giordana, A. (1998). A knowledge intensive approach to concept induction. In J. E. Laird (Ed.), *Proceedings of the Fifth International Conference on Machine Learning* (pp. 305–317). Morgan Kaufmann.

Bergmann, R. (1998). On the use of taxonomies for representing case features and local similarity measures. In L. Gierl & M. Lenz (Eds.), *Proceedings of the Sixth German Workshop on CBR* (pp. 23–32).

Bergmann, R. (2002). *Experience management.* Berlin: Springer.

Bergmann, R., Breen, S., Goeker, M., Managao, M., & Wess, S. (1999). Developing Industrial Case-Based Reasoning Applications: The INRECA Methodology. In *Lnai 1612.* Berlin: Springer.

Bergmann, R., Pews, G., & Wilke, W. (1994). Explanation-based similarity:

A unifying approach for integrating domain knowledge into Case-Based Reasoning. In S. Wess, K.-D. Althoff, & M. M. Richter (Eds.), *Topics in Case-Based Reasoning* (p. 182-196). Springer.

Bergmann, R., & Vollrath, I. (1999). Generalized cases: Representation and steps towards efficient similarity assessment. In W. Burgard, T. Christaller, & A. B. Cremers (Eds.), *KI-99: Advances in Artificial Intelligence* (p. 195-206).

Bergmann, R., Vollrath, I., & Wahlmann, T. (1999). Generalized cases and their application to electronic designs. In E. Melis (Ed.), *Proceedings of the 7th German Workshop on CBR* (pp. 6–19).

Bergmann, R., & Wilke, W. (1998). Towards a new formal model of transformational adaptation in Case-Based Reasoning. In H. Prade (Ed.), *ECAI98, Thirteenth European Conference on Artificial Intelligence* (pp. 53–57). Chichester: John Wiley and Sons.

Bergmann, R., Wilke, W., Vollrath, I., & Wess, S. (1996). Integrating general knowledge with object-oriented case representation and reasoning. In H.-D. Burkhard & M. Lenz (Eds.), *4th German Workshop: Case-Based Reasoning - System Development and Evaluation, Informatik-Berichte Nr. 55* (p. 120-127). Berlin: Humboldt-Universitaet.

Blake, C. L., & Merz, C. J. (1998). *UCI repository of machine learning databases.* (http://www.ics.uci.edu/∼mlearn/MLRepository.html)

Boer, R. de, Kok, J., & Groen, F. C. A. (2002). Uva trilearn 2001 team description. In A. Birk, S. Coradeschi, & S. Tadokoro (Eds.), *RoboCup 2001: Robot Soccer World Cup V. Lecture Notes in computer science 2377* (pp. 551–554). Berlin: Springer.

Boerner, K. (1994). Term-based approach to structural similarity as guidance for adaptation. In A. Voss (Ed.), *FABEL - Similarity concepts and retrieval methods* (pp. 59–72). Sankt Augustin: GMD.

Bonzano, A., Cunningham, P., & Smyth, B. (1997). Using introspective learning to improve retrieval in cbr: A case study in air traffic control. In D. Leake & E. Plaza (Eds.), *Proceedings of the Second ICCBR conference* (p. 291-302). Berlin: Springer.

Branting, K. (1989). Integrating generalizations with exemplar-based reasoning. In G. M. Olson & E. E. Smith (Eds.), *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society* (pp. 139–146). Hillsdale, New Jersey: Lawrence Erlbaum.

Branting, K., & Porter, B. W. (1991). Rules and precedents as complementary warrants. In *National Conference on Artificial Intelligence* (Vol. 1,

p. 3-9). Menlo Park, CA: AAAI.

Brown, A. L. (1989). Analogical mapping and transfer: What develops? In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning* (pp. 369–412). Cambridge: Cambridge University Press.

Bruner, J. S., Goodnow, J. J., & Austin, G. A. (1956). *A study of thinking.* John Wiley & Sons, Inc.

Bunke, H., & Messmer, B. T. (1994). Similarity measures for structured representations. In S. Wess, K.-D. Althoff, & M. M. Richter (Eds.), *Topics in CBR: EWCBR-93 - First European Workshop on Case-Based Reasoning* (pp. 77–91). Berlin: Springer.

Burkhard, H.-D. (1998). Extending some Concepts of CBR - Foundations of Case Retrieval Nets. In M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard, & S. Wess (Eds.), *Case-Based Reasoning Technology, From Foundations to Applications* (p. 17-50). Berlin: Springer.

Buttinger, S., Diedrich, M., Hennig, L., Hoenemann, A., Huegelmeyer, P., Nie, A., et al. (2001). *ORCA project report* (Tech. Rep.). University of Osnabrueck.

Cain, T., Pazzani, M. J., & Silverstein, G. (1991). Using domain knowledge to influence similarity judgements. In *Proceedings of the Case-Based Reasoning Workshop* (pp. 191–198). Washington D.C., U.S.A.

Carey, S. (1984). Are children fundamentally different kinds of thinkers and learners than adults? In S. F. Chipman, J. W. Segal, & R. Glaser (Eds.), *Thinking and learning skills: Current research and open questions* (pp. 485–517). Hillsdale, New Jersey: Erlbaum.

Carmel, D., & Markovich, S. (1996). Learning models of intelligent agents. In H. Shrobe & T. Senator (Eds.), *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth innovative applications of Artificial Intelligence Conference, vol. 2* (pp. 62–67). Menlo Park, California: AAAI Press.

Carnap, R. (1928). *Der logische Aufbau der Welt.* Berlin: Weltkreis-Verlag.

Chen, M., Foroughi, E., Heintz, F., Huang, Z., Kapetanakis, S., Kostiadis, K., et al. (2001). *Soccerserver manual v7.*

Chi, M. T. H., Feltovich, P. J., & Glaser, R. (1981). Categorization and representation of physics problems by experts and novices. *Cognitive Science*, *5*, 121–152.

Choplin, J., Cheng, P., & Holyoak, K. (2001). Causal information as a constraint on similarity. In J. D. Moore (Ed.), *Proceedings of the 23rd Annual Conference of the Cognitive Science Society* (pp. 212–216). Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Clark, P. (1989). *Exemplar-based reasoning in geological prospect appraisal* (Tech. Rep. No. 89-034). Glasgow: Turing Institute.

Coulon, C.-H., & Steffens, R. (1994). Comparing fragments by their images. In A. Voss (Ed.), *FABEL - Similarity concepts and retrieval methods* (pp. 36–44). Sankt Augustin: GMD.

Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*, 21–27.

Craw, S. (2003). Introspective learning to build Case-Based Reasoning (CBR) knowledge containers. In P. Perner & A. Rosenfeld (Eds.), *Proceedings of the Third International Conference on Machine Learning and Data Mining in Pattern Recognition* (pp. 1–6). Berlin: Springer.

Creecy, R. H., Masand, B. M., Smith, S. J., & Waltz, D. L. (1992). Trading mips and memory for knowledge engineering. *Communications of the ACM*, *35*(8), 48–64.

Daelemans, W., & Bosch, A. van den. (1992). Generalization performance of backpropagation learning on a syllabification task. In *Proceedings of the Third Twente Workshop on Language Technology: Connectionism and Natural Language Processing* (pp. 27–37). Enschede, The Netherlands: Unpublished.

DeJong, G. (1989). The role of explanation in analogy; or, the curse of an alluring name. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning* (pp. 346–365). New York, NY: Cambridge University Press.

Denzinger, J., & Hamdan, J. (2004). Improving modeling of other agents using stereotypes and compactification of observations. In *Proceedings of 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)* (pp. 1414–1415). IEEE Computer Society.

Diaz-Agudo, B., & Gonzalez-Calero, P. A. (2001). Knowledge intensive cbr made affordable. In A. Aamodt, D. Patterson, & B. Smyth (Eds.), *Proceedings of the Workshop program at the Fourth International Conference on Case-Based Reasoning.*

Domingos, P. (1995). Rule induction and instance-based learning: A unified approach. In *Proceedings of the Fourteenth International Joint Conference on Articial Intelligence (IJCAI)* (p. 1226-1232). Morgan Kaufmann.

Domingos, P. (1997). Context-sensitive feature selection for lazy learners. *Artificial Intelligence Review*, *11*, 227–253.

Dylla, F., Ferrein, A., Lakemeyer, G., Murray, J., Obst, O., Roefer, T., et al. (2005). Towards a league-independent qualitative soccer theory for robocup. In D. Nardi, M. Riedmiller, & C. Sammut (Eds.), *RoboCup-2004: The Seventh RoboCup Competitions and Conferences* (p. 611-619). Berlin: Springer.

Erickson, M. A., & Kruschke, J. K. (1998). Rules and exemplars in category learning. *Journal of Experimental Psychology: General*, *127*, 107–140.

Fawcett, T. E., & Utgoff, P. E. (1992). Automatic feature generation for problem solving systems. In D. H. Sleeman & P. Edwards (Eds.), *Proceedings of the 9th International Conference on Machine Learning* (pp. 144–153). Morgan Kaufmann.

Fayyad, U. M., & Irani, K. B. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. In R. Bajcsy (Ed.), *Proceedings of the 13th International Joint Conference on Artificial Intelligence* (pp. 1022–1027). San Mateo, CA: Morgan Kaufmann.

Friedman, J. H. (1994). *Flexible metric nearest neighbor classification* (Tech. Rep. No. 113). Stanford University Statistics Department. (available from http://citeseer.ist.psu.edu/friedman94flexible.html)

Fu, L.-M., & Buchanan, B. G. (1985). Learning intermediate concepts in constructing a hierarchical knowledge base. In A. Joshi (Ed.), *Proceedings of the Ninth International Joint Conference on Artificial Intelligence* (pp. 659–666). Morgan Kaufmann.

Fullerton, G. S. (1890). *On sameness and identity.* Philadelphia: University of Pennsylvania Press.

Gabel, T., & Stahl, A. (2004). Exploiting background knowledge when learning similarity measures. In *Proceedings of the ECCBR 2004.*

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, *7*, 155–170.

Gentner, D. (1988). Metaphor as structure mapping: The relational shift. *Child Development*, *59*, 47–59.

Gentner, D. (1989). The mechanisms of analogical learning. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning* (pp. 199–241). Cambridge: Cambridge University Press.

Goodman, N. (1951). *The structure of appearance.* Cambridge, MA: Harvard University Press.

Goodman, N. (1972). Seven strictures on similarity. In N. Goodman (Ed.), *Problems and projects* (pp. 437–447). Indianapolis and New York: The BobbsMerrill Company.

Griffiths, A. D., & Bridge, D. (1997). Towards a theory of optimal similarity measures. In N. Filer & I. Watson (Eds.), *Proceedings of the Third UK Workshop on Case-Based Reasoning.*

Griffiths, A. D., & Bridge, D. G. (1996). A yardstick for the evaluation of case-based classifiers. In I. D. Watson (Ed.), *Proceedings of Second UK Workshop on Case-Based Reasoning.*

Groot, A. D. de. (1978). *Thought and choice in chess* (2nd ed.). The Hague: Mouton.

Guarino, N., & Giaretta, P. (1995). Ontologies and knowledge bases: Towards a terminological clarication. In N. Mars (Ed.), *Towards very large knowledge bases* (pp. 25–32). Amsterdam: IOS Press.

Gunsch, M. G. H., & Rendell, L. A. (1991). Opportunistic constructive induction: Using fragments of domain knowledge to guide construction. In L. A. Birnbaum & G. C. Collins (Eds.), *Proceedings of the Eighth International Workshop on Machine Learning* (pp. 147–152). Morgan Kaufmann.

Hahn, U., & Chater, N. (1998). Understanding similarity: A joint project for psychology, Case-Based Reasoning and law. *Artificial Intelligence Review*, *12*, 393–427.

Hahn, U., Chater, N., & Richardson, L. B. (2003). Similarity as transformation. *Cognition*, *87*, 1–32.

Hume, D. (1777/1975). *An enquiry concerning human understanding.* Indianapolis: Hackett Publishing Company.

James, W. (1890). *The principles of psychology.* New York: Henry Holt.

Janetzko, D., Wess, S., & Melis, E. (1992). Goal-driven similarity assessment. In H. J. Ohlbach (Ed.), *GWAI-92 16th German Workshop on Artificial Intelligence.* Springer.

Jansen, P. (1990). Problematic positions and speculative play. In T. A. Marsland & J. Schaeffer (Eds.), *Computers, chess and cognition* (pp. 169–182). New York: Springer.

Jurisica, I. (1994). *Context-based similarity applied to retrieval of relevant cases* (Tech. Rep. No. DKBS-TR-94-5). Toronto, Ontario: University of Toronto, Department of Computer Science.

Kaminka, G. A., & Avrahami, D. (2004). Symbolic behavior-recognition. In M. Bauer, P. Gmytrasiewicz, G. A. Kaminka, & D. V. Pynadath (Eds.), *Workshop on modeling other agents from observations at aamas 2004* (pp. 73–80).

Kautz, H. (1991). A formal theory of plan recognition and its implementation. In J. Allen, H. Kautz, R. Pelavin, & J. Tenenberg (Eds.), *Reasoning about plans* (pp. 69–125). San Mateo, CA: Morgan Kaufman.

Kira, K., & Rendell, L. A. (1992). A practical approach to feature selection. In D. H. Sleeman & P. Edwards (Eds.), *Proceedings of the Ninth International Workshop on Machine Learning* (pp. 249–256). Morgan Kaufmann Publishers Inc.

Kitano, H., Tambe, M., Stone, P., Veloso, M., Coradeschi, S., Osawa, E., et al. (1997). The robocup synthetic agent challenge,97. In *International Joint Conference on Artificial Intelligence (IJCAI97)* (pp. 24–29). San Francisco, CA: Morgan Kaufmann.

Klinkenberg, R. (1998). *Maschinelle Lernverfahren zum adaptiven Informationsfiltern bei sich veraendernden Konzepten.* Unpublished master's thesis, Fachbereich Informatik, Universitaet Dortmund, Germany.

Knauff, M., & Schlieder, C. (1994). Dynamic grouping: case reinterpretation as a foundation of knowledge-intensive similarity assessment. In A. Voss (Ed.), *FABEL - Similarity concepts and retrieval methods* (pp. 85–104). Sankt Augustin: GMD.

Kolodner, J. (1991). Improving human decision making through case-based decision aiding. *AI Magazine*, *12*(2), 52–68.

Kolodner, J. (1993). *Case-Based Reasoning.* San Mateo: Morgan Kaufmann.

Kononenko, I. (1994). Estimating attributes: Analysis and extensions of RELIEF. In F. Bergadano & L. de Raedt (Eds.), *Proceedings of the European Conference on Machine Learning* (p. 171-182). Berlin: Springer.

Koton, P. (1988). Reasoning about evidence in causal explanations. In *Proceedings of the Seventh National Conference on Artificial Intelligence (aaai-88)* (pp. 256–261). Morgan Kaufmann.

Kuhlmann, G., Stone, P., & Lallinger, J. (2005). The ut austin villa 2003 champion simulator coach: A machine learning approach. In D. Nardi, M. Riedmiller, C. Sammut, & J. Santos-Victor (Eds.), *RoboCup 2004: Robot Soccer World Cup VIII* (Vol. 3276, p. 636-644). Berlin: Springer.

Lamberts, K. (1994). Flexible tuning of similarity in exemplar-based categorization. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *20*, 1003–1021.

Lamberts, K., & Chong, S. (1998). Dynamics of dimension weight distribution and flexibility in categorization. In M. Oaksford & N. Chater (Eds.), *Rational models of cognition* (pp. 275–292). Oxford University Press.

Lamma, E., Riguzzi, F., & Storari, S. (2004). Exploiting association and correlation rules - parameters for improving the k2 algorithm. In R. L. de Mantaras & L. Saitta (Eds.), *Proceedings of ECAI 2004* (pp. 500–504). IOS Press.

Langley, P., & Iba, W. (1993). Average-case analysis of a nearest neighbor algorithm. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence* (pp. 889–894). San Mateo: Morgan Kaufmann.

Leake, D. B., & Wilson, D. C. (1999). Combining CBR with interactive knowledge acquisition, manipulation and reuse. In K.-D. Althoff, R. Bergmann, & K. Branting (Eds.), *Proceedings of the Third International Conference on Case-Based Reasoning* (pp. 203–217). Berlin: Springer-Verlag.

Ledezma, A., Aler, R., Sanchís, A., & Borrajo, D. (2005). Predicting opponent actions by observation. In D. Nardi, M. Riedmiller, C. Sammut, & J. Santos-Victor (Eds.), *RoboCup 2004: Robot Soccer World Cup VIII* (Vol. 3276, p. 286-296). Berlin: Springer.

Ling, C. X., Parry, J. J., & Wang, H. (1997). Setting attribute weights for nearest neighbour learning algorithms using C4.5. *International Journal of Pattern Recognition and Artificial Intelligence*, *11*(3), 405 – 415.

Ling, C. X., & Wang, H. (1997). Computing optimal attribute weight settings for nearest neighbour algorithms. *Artificial Intelligence Review*, *11*, 255–272.

Manning, C. D., & Schuetze, H. (1999). *Foundations of statistical natural language processing*. Cambridge, Massachusetts: The MIT Press.

Markman, A. B. (2001). Structural alignment, similarity, and the internal structure of category representations. In U. Hahn & M. Ramscar (Eds.), *Similarity and categorization* (pp. 109–130). Oxford: Oxford University Press.

Markman, A. B., & Gentner, D. (1990). Analogical mapping during similarity judgments. In *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society* (pp. 38–44). Cambridge, MA: Lawrence Erlbaum Associates.

Markman, A. B., & Gentner, D. (2005). Nonintentional similarity processing. In R. R. Hassin, J. A. Bargh, & J. S. Uleman (Eds.), *The new unconscious* (pp. 107–137). New York: Oxford University Press.

Marling, C., Tomko, M., Gillen, M., Alexander, D., & Chelberg, D. (2003). Case-Based Reasoning for planning and world modeling in the RoboCup small sized league. In U. Visser (Ed.), *IJCAI Workshop on issues in designing physical agents for dynamic real-time environments.*

Matheus, C. J. (1991). The need for constructive induction. In L. A. Birnbaum & G. C. Collins (Eds.), *Proceedings of the Eighth International Workshop on Machine Learning* (pp. 173–177). Morgan Kaufmann.

Medin, D. L., Goldstone, R. L., & Gentner, D. (1993). Respects for similarity. *Psychological Review*, *100*(2), 254–278.

Medin, D. L., & Ortony, A. (1989). What is psychological essentialism? In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning* (pp. 179–195). New York, NY: Cambridge University Press.

Mehlhorn, K. (1984). *Graph algorithms and NP-completeness.* Berlin: Springer.

Mooney, R. J., & Ourston, D. (1991). Constructive induction in theory refinement. In L. Birnbaum & G. Collins (Eds.), *Proceedings of the Eighth International Machine Learning Workshop* (pp. 178–182). San Mateo, CA: Morgan Kaufmann.

Motta, E., Fensel, D., Gaspari, M., & Benjamins, R. (1999). Specifications of knowledge components for reuse. In *Proceedings of the 11th International Conference on Software Engineering and Knowledge Engineering, SEKE'99* (pp. 36–43). KSI Press.

Newell, A., & Simon, H. (1990). Computer science as empirical study: symbols and search. In M. A. Boden (Ed.), *The philosophy of Artificial Intelligence.* Oxford, UK: Oxford University Press.

Nosofsky, R. M. (1984). Choice, similarity, and the context theory of classification. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *115*, 39–57.

Nosofsky, R. M. (1990). *Exemplars, prototypes, and similarity rules* (Tech. Rep. No. 17). Bloomington, IN: Indiana University, Department of Psychology.

Nosofsky, R. M., Palmeri, T. J., & McKinley, S. (1994). Rule-plus-exception model of classication learning. *Psychological Review*, *101*, 53–79.

Nunez, H., Sanchez-Marre, M., Cortes, U., Comas, J., Rodriguez-Roda, I., & Poch, M. (2002). Feature weighting techniques for prediction tasks in environmental processes. In *Proceedings of the 3rd Workshop on Binding Environmental Sciences and Artificial Intelligence (BESAI 2002).*

Okamoto, S., & Yugami, N. (2000). Generalized average-case analysis of the nearest neighbor algorithm. In P. Langley (Ed.), *Proceedings of the*

Seventeenth International Conference on Machine Learning (ICML) (pp. 695–702). Morgan Kaufmann.

Okamoto, S., & Yugami, N. (2003). Effects of domain characteristics on instance-based learning algorithms. *Theoretical Computer Science*, *298*(1), 207–233.

Pavlov, I. P. (1927). *Conditioned reflexes.* London, UK: Oxford University Press.

Porter, B. W., Bareiss, R., & Holte, R. C. (1990). Concept learning and heuristic classification in weak-theory domains. *Artificial Intelligence*, *45*(1-2), 229-263.

Quinlan, R. (1993). *C4.5: Programs for Machine Learning.* San Mateo, CA: Morgan Kaufmann.

Ragavan, H., & Rendell, L. (1991). Relations, knowledge and empirical learning. In L. A. Birnbaum & G. C. Collins (Eds.), *Proceedings of the Eighth International Workshop on Machine Learning* (pp. 188–192). Morgan Kaufmann.

Rendell, L. A. (1989). Comparing systems and analyzing functions to improve constructive induction. In A. M. Segre (Ed.), *Proceedings of the Sixth International Workshop on Machine Learning* (pp. 461–464). Morgan Kaufmann.

Ricci, F., & Avesani, P. (1995). Learning a local similarity metric for Case-Based Reasoning. In *Proceedings of the First International Conference on Case-Based Reasoning* (p. 301-312). Springer.

Richter, M. M. (1992). Classification and learning of similarity measures. In O. Opitz, B. Lausen, & R. Klar (Eds.), *Proceedings der Jahrestagung der Gesellschaft fuer Klassifikation.* Springer.

Richter, M. M. (1995). *The knowledge contained in similarity measures.* Invited talk at ICCBR-95.

Richter, M. M. (2003). Fallbasiertes Schliessen. *Informatik Spektrum*, *3*(26), 180–190.

Richter, M. M., & Althoff, K.-D. (1999). Similarity and utility in non-numerical domains. In W. G. und Martin Schader (Ed.), *Mathematische Methoden der Wirtschaftswissenschaften* (pp. 403 – 413). Physika-Verlag.

Riedmiller, M., Merke, A., Nowak, W., Nickschas, M., & Withopf, D. (2003). Brainstormers 2003 - team description. In D. Polani, A. Bonarini, B. Browning, & K. Yoshida (Eds.), *Pre-proceedings of robocup 2003.*

Riley, P. (2005). *Coaching: Learning and Using Environment and Agent Mod-*

els for Advice. Unpublished doctoral dissertation, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA.

Riley, P., & Veloso, M. (2002). Planning for distributed execution through use of probabilistic opponent models. In M. Ghallab, J. Hertzberg, & P. Traverso (Eds.), *Proceedings of the Sixth International Conference on AI Planning and Scheduling (AIPS-2002)* (pp. 72–81). Menlo Park, CA: AAAI Press.

Rips, L. J. (1989). Similarity, typicality, and categorization. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning* (pp. 21–59). New York: Cambridge University Press.

Rissland, E. L., & Skalak, D. B. (1989). Combining case-based and rule-based reasoning: A heuristic approach. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (pp. 524–530). San Mateo, CA: Morgan Kaufmann.

Rodriguez, A. R. (2001). Issues in case-based reasoning. In U. Hahn & M. Ramscar (Eds.), *Similarity and categorization* (pp. 131–154). Oxford: Oxford University Press.

Rogowski, C. (2004). Model-based opponent-modelling in domains beyond the prisoner's dilemma. In M. Bauer, P. Gmytrasiewicz, G. A. Kaminka, & D. V. Pynadath (Eds.), *Workshop on modeling other agents from observations at aamas 2004* (pp. 41–48).

Rosch, E., & Mervis, C. B. (1975). Family resemblances: Studies in the internal structure of categories. *Cognitive Psychology*, *7*, 573–605.

Ross, B. (1989). Some psychological results on case-based reasoning. In K. Hammond (Ed.), *Proceedings of the DARPA Case-Based Reasoning Workshop* (pp. 144–147). San Mateo: Morgan Kaufmann.

Salzberg, S. (1991). A nearest hyperrectangle learning method. *Machine Learning*, *6*(3), 251–276.

Schaaf, J. W. (1994). Detecting gestalts in CAD-plans to be used as indices. In A. Voss (Ed.), *FABEL - Similarity concepts and retrieval methods* (pp. 73–84). Sankt Augustin: GMD.

Schmid, U., Wirth, J., & Polkehn, K. (2003). A closer look at structural similarity in analogical transfer. *Cognitive Science Quarterly*, *3*(1), 57–89.

Schreiber, G., Wielinga, B. J., & Jansweijer, W. H. J. (1995). The KACTUS view on the 'O' word. In J. C. Bioch & Y.-H. Tan (Eds.), *Proceedings of the 7th Dutch National Conference on Artificial Intelligence NAIC'95* (p. 159—168). Erasmus University Rotterdam, The Netherlands: EU-

RIDIS.

Shanks, D. R. (1995). *The psychology of associative learning.* Cambridge, England: Cambridge University Press.

Shannon, C. E. (1950). Programming a computer for playing chess. *Philosophical Magazine*, *41*, 256–275.

Shepard, R. N. (1957). Stimulus and response generalization: a stochastic model relating generalization to distance in psychological space. *Psychometrika*, *22*, 325–345.

Smith, E. E., Patalano, A. L., & Jonides, J. (1998). Alternative strategies of categorization. *Cognition*, *65*, 167–196.

Smith, L. B. (1989). From global similarity to kinds of similarity: The construction of dimensions in development. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning* (pp. 146–178). Cambridge: Cambridge University Press.

Spanoudakis, G., & Constantopoulos, P. (1994). Measuring similarity between software artifacts. In *Proceedings of the 6th International Conference on Software Engineering and Knowledge Engineering, SEKE'94.* Skokie: Knowledge Systems Institute.

Stahl, A. (2004). *Learning of Knowledge-Intensive Similarity Measures in Case-Bases Reasoning* (Vol. 986). Verlag dissertation.de. (Dissertation an der Technischen Universitaet Kaiserslautern)

Steffens, T. (2002). *Feature-based declarative opponent-modelling in multiagent systems.* Unpublished master's thesis, Institute of Cognitive Science Osnabrueck.

Steffens, T. (2004a). Adapting similarity-measures to agent-types in opponent-modelling. In M. Bauer, P. Gmytrasiewicz, G. A. Kaminka, & D. V. Pynadath (Eds.), *Workshop on Modeling Other Agents from Observations at AAMAS 2004* (pp. 125–128).

Steffens, T. (2004b). Similarity-measures based on imperfect domain-theories. In S. Staab & E. Onainda (Eds.), *Proceedings of STAIRS 2004* (pp. 193–198). IOS Press, Frontiers in Artificial Intelligence and Applications.

Steffens, T. (2004c). Virtual attributes from imperfect domain theories. In B. Lees (Ed.), *Proceedings of the 9th UK Workshop on Case-Based Reasoning at AI-2004* (pp. 21–29).

Steffens, T. (2005a). Knowledge-intensive similarity-based opponent modelling. In D. W. Aha (Ed.), *Proceedings of the IJCAI Workshop on representation, reasoning, and learning in computer games.*

Steffens, T. (2005b). Knowledge-rich similarity-based classification. In H. Munoz-Avila (Ed.), *Proceedings of the International Conference on Case-Based Reasoning (ICCBR).*

Steffens, T. (2005c). Partial and vague knowledge for similarity measures. In L. P. Kaelbling & A. Saffiotti (Eds.), *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI).* Denver: Gallup House.

Steffens, T. (2005d). Similarity-based opponent modelling using imperfect domain theories. In G. Kendall & S. Lucas (Eds.), *IEEE 2005 Symposium on Computational Intelligence and Games (CIG'05)* (pp. 285–291). Colchester, UK: Essex University.

Stepp, R. E., & Michalski, R. S. (1986). Conceptual clustering: Inventing goal-oriented classifications of structured objects. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine Learning: An Artificial Intelligence Approach* (Vol. II). Los Altos, CA: Morgan Kaufman Publishers, Inc.

Stolzenburg, F., Murray, J., & Sturm, K. (2003). Multiagent matching algorithms with and without coach. In M. Schillo, M. Klusch, J. Mueller, & H. Tianfield (Eds.), *Proceedings of the 1st German Conference on multiagent system technologies* (pp. 192–204). Berlin: Springer.

Stone, P., Riley, P., & Veloso, M. M. (2000). Defining and using ideal teammate and opponent agent models. In H. Kautz & B. Porter (Eds.), *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAA)* (p. 1040-1045). Menlo Park, CA: AAAI Press / The MIT Press.

Strube, G., Enzinger, A., Janetzko, D., & Knauff, M. (1995). Knowledge engineering for cbr systems from a cognitive science perspective. In M. M. Veloso & A. Aamodt (Eds.), *ICCBR '95: Proceedings of the First International Conference on Case-Based Reasoning Research and Development* (pp. 548–558). London, UK: Springer-Verlag.

Surma, J. (1994). Enhancing similarity measure with domain specific knowledge. In *Proceedings of the Second European Conference on Case-Based Reasoning* (pp. 365–371). Paris: AcknoSoft Press.

Tartakovski, A., & Maximini, R. (2003). Similarity assessment and retrieval of generalized cases. In R. Bergmann & M. Schaaf (Eds.), *Proceedings of the Workshop on knowledge and experience management.*

Ting, K. M. (1997). Discretisation in lazy learning algorithms. *Artificial*

Intelligence Review, *11*, 157–174.

Towell, G. G., Shavlik, J. W., & Noordenier, M. O. (1990). Refinement of approximate domain theories by knowledge based neural network. In *Proceedings of the Eighth National Conference on AI* (Vol. 2, pp. 861–866).

Turney, P. (1996). The management of context-sensitive features: A review of strategies. In *Proceedings of the Workshop on Learning in Context-sensitive Domains at the 13th International Conference on Machine Learning* (pp. 60–65).

Tversky, A. (1977). Features of similarity. *Psychological Review*, *84*(4), 327–352.

Visser, U., Drücker, C., Hübner, S., Schmidt, E., & Weland, H.-G. (2001). Recognizing formations in opponent teams. In P. Stone, T. R. Balch, & G. K. Kraetzschmar (Eds.), *RoboCup 2000: Robot Soccer World Cup IV* (Vol. 2019, p. 391-396). Berlin: Springer.

Visser, U., & Weland, H.-G. (2003). Using online learning to analyze the opponent's behavior. In G. A. Kaminka, P. U. Lima, & R. Rojas (Eds.), *RoboCup 2002: Robot Soccer World Cup VI* (Vol. 2752, p. 78-93). Berlin: Springer.

Wallach, M. A. (1958). On psychological similarity. *Psychological Review*, *65*, 103–116.

Weiss, G. (1999). *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence.* Cambridge, Massachusetts: MIT Press.

Weiss, S. M., & Kulikowski, C. A. (1991). *Computer systems that learn.* San Mateo, CA: Morgan Kaufmann.

Wendler, J. (2004). Recognizing and predicting agent behavior with case-based reasoning. In D. Polani, A. Bonarini, B. Browning, & K. Yoshida (Eds.), *RoboCup 2003: Robot Soccer World Cup VII, Lecture Notes in Artificial Intelligence* (p. 729-738). Berlin: Springer.

Wertheimer, M. (1923). Untersuchungen zur Lehre von der Gestalt. II. *Psychologische Forschung*, *4*, 301–350.

Wess, S., & Globig, C. (1994). Case-based and symbolic classification algorithms - a case study using version space. In S. Wess, K.-D. Althoff, & M. M. Richter (Eds.), *Topics in CBR: EWCBR-93 - First European Workshop on Case-Based Reasoning* (pp. 77–91). Berlin: Springer.

Wettschereck, D., Aha, D. W., & Mohri, T. (1997). A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review*, *11*, 273–314.

Wilke, W., & Bergmann, R. (1996). Adaptation with the INRECA system. In A. Voss (Ed.), *Proceedings of the ECAI 96 Workshop: Adaptation in CBR.*

Wilke, W., & Bergmann, R. (1998). Techniques and knowledge used for adaptation during case-based problem solving. In *Proceedings of the 11th International Conference on industrial and engineering applications of Artificial Intelligence and expert systems* (Vol. 2, pp. 497–506). Berlin: Springer.

Wilke, W., Vollrath, I., Althoff, K.-D., & Bergmann, R. (1997). A framework for learning adaptation knowledge based on knowledge-light approaches. In R. Bergmann & W. Wilke (Eds.), *Proceedings of the 5th German Workshop on Case-Based Reasoning* (pp. 235–242). Centre for Learning Systems and Applications, University of Kaiserslautern.

Wogulis, J., & Langley, P. (1989). Improving efficiency by learning intermediate concepts. In N. S. Sridharan (Ed.), *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (pp. 657–662). Los Altos, CA: Morgan Kaufmann.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Information Retrieval*, *1*(1/2), 69–90.