Learning in Case-Based Classification Algorithms*

Christoph Globig, Stefan Wess
University of Kaiserslautern, P.O. Box 3049
D-67653 Kaiserslautern, Germany
{globig,wess}@informatik.uni-kl.de

Abstract

While symbolic learning approaches encode the knowledge provided by the presentation of the cases *explicitly* into a *symbolic representation of the concept*, e.g. formulas, rules, or decision trees, case-based approaches describe learned concepts *implicitly* by a pair (CB, d), i.e. by a set CB of cases and a distance measure d. Given the same information, symbolic as well as the case-based approach compute a classification when a new case is presented. This poses the question if there are any differences concerning the learning power of the two approaches. In this work we will study the relationship between the case base, the measure of distance, and the target concept of the learning process. To do so, we transform a simple symbolic learning algorithm (the version space algorithm) into an equivalent case-based variant. The achieved results strengthen the conjecture of the equivalence of the learning power of symbolic and case-based methods and show the interdependency between the measure used by a case-based algorithm and the target concept.

1 Introduction

In this paper which is an extended version of (Wess & Globig, 1994) we want to compare two important learning paradigms – the *symbolic* (Michalski, Carbonell, & Mitchell, 1983) and the *case-based* approach (Aha, 1991). As a first step in this direction, Jantke (1992) has already analyzed the common points of inductive inference and case-based learning. The learning task we study is the classification of objects (cases). The aim of a classification task is to map the objects $\mathbf{x}$ of a universe U to certain concepts $C \subseteq U$, i.e. to subsets of the universe. In the most simple scenario we have to decide the membership problem of a concept C, i.e. the universe U is separated in two disjoint subsets C and $\neg C$.

*The presented work was partly supported by the *Deutsche Forschungsgemeinschaft*, project IND-CBL.

We present a simple symbolic learning algorithm (the Version Space (Mitchell, 1982)) and transform it into a case-based variant. Based on this example we will show that for case-based approaches there exists a strong tradeoff between the set of representable concepts and the minimal number of cases in the case base. Thus for our scenario the used bias must have a comparable strength in both approaches.

The second important component of a case-based learning system is the case selection strategy, i.e. the method to select appropriate cases for the case base. We study different types of case selection strategies and elaborate relations between the corresponding case-based learning types and relate them to Gold-style language learning (cf. (Gold, 1967)) from positive and both positive and negative examples.

1.1 Symbolic Learning

Under the term *symbolic learning*[1] we subsume approaches, e.g. (Michalski et al., 1983), that code the knowledge provided by the presentation of the cases into a *symbolic representation of the concept* only, e.g. by formulas, rules, or decision trees. These learning approaches produce after each presentation of a case a hypothesis formulated in a pregiven (formal) hypothesis language. The aim is to converge against a hypothesis that fulfills a pregiven criterion of correctness.

We will call the phase while the algorithms build their hypothesis *learning phase* and the phase while these hypotheses are used to classify new objects *application phase*. The fundamental problem the symbolic and the case-based approach have to solve during the learning phase is the same. At every moment the system knows the correct classification of a finite subset of the universe only. The knowledge that the algorithm is able to use is incomplete and, therefore, the computed hypothesis needs not to be correct.

Symbolic approaches can be characterized along the following dimensions (Jantke & Lange, 1989):

Problem class: For the characterization of a certain algorithm it is important to know the class of problems that it has to solve.

Presented information: The learner may get information of different types. It is important to specify, how the presented cases are selected. They may follow an enumeration of all objects or they are drawn according to a given distribution.

Semantic of the presented information: The relation between the presented information and the problem class must be specified.

[1] Case-based systems may also use symbolic knowledge. The use of the term "symbolic learning" in this work may therefore be confusing to the reader. But, since the term "symbolic learning" is also used to contrast a special class of learning approaches to systems which use neural networks, we think that the use of the term "symbolic learning" as characterization of these approaches is appropriate.

Hypothesis space: Like the problem class the class of allowed hypotheses must be characterized. To represent their hypotheses symbolic algorithms may for example use a fragment of the predicate logic.

Learning algorithms: For a description of the learning problem the set of allowed learning algorithms must be given. We will demand that the algorithms produce a hypothesis after each presentation.

Convergence of the sequence of hypotheses: We have assumed that the learning algorithm produces a hypothesis after each presentation. It must be clarified which hypothesis is identified by a given sequence in the limit.

Successful learning: Because of the learning system produces a sequence of hypotheses, there must be a criterion, whether this sequence is a successful learning.

It is important to remember that in symbolic learning cases are not used during the application phase, i.e. to classify new objects. The knowledge provided by the presentation of the cases during the learning phase is completely coded in the symbolic representation of the hypothesis. This compilation process may be seen as a abstraction step.

These terms could also be used to describe case-based algorithms. However, case-based systems have additional characteristical properties that we will describe in the next section.

2 Case-Based Learning

Case-based learning (Aha, 1991) applies techniques of nearest neighbor classification (Dasarathy, 1990) in symbolic domains. In case-based learning the learning phase and the application phase are not strictly separated (Kolodner, 1993). The basic idea is to use the knowledge of the known cases *directly* to solve new problems. Direct means that the system does not try to extract the whole knowledge from the case to operate with the extracted knowledge only. All cases (or a subset) are stored and interpreted during the solution of new problems.

Case-based methods solve a given problem in a sequence of steps (Aamodt & Plaza, 1994):

Retrieve: In a first step the system tries to retrieve the relevant cases from the case base CB. From the cases retrieved in the first step the learner has to select one. We will call this case the *reference case*.

Reuse: The reference cases is used to solve the new problem. If the case does not match perfectly the new situation, it has to be adapted. Therefore, the learner must have knowledge which modifications are allowed. This knowledge is tightly related to the domain, where the learning takes place. The complexity of modifications vary from simple parameter changes to the construction of new solutions for parts of the old solution that cannot be reused in the new situation.

Revise: If the learner has the ability to evaluate the new solution, a test of the solution will follow. If there arise some problems, the solution must be modified again.

Retain: In the learning phase the learner may change its knowledge depending of the feedback of the user. Learning may change all the components of the learner. The easiest way of modification relates to the case base. The new solution can be stored in the case base. If the reference case was not optimal, the methods for the retrieval may be changed.

The following section describes the basic algorithm we want to use for our learning problem.

2.1 Basic Algorithm

In the application phase, a case-based system tries to classify a new case with respect to a set of stored cases, the case base CB. For simplicity, we consider cases as tuples $(\mathbf{x}, class(\mathbf{x}))$ where $\mathbf{x}$ is a description of the case and $class(\mathbf{x})$ is the classification. Given a new case $(\mathbf{y}, ?)$ with unknown classification, the system searches in the case base CB for the nearest neighbor $(\mathbf{x}, class(\mathbf{x}))$ (or the most similar case) according to a given distance measure d. Then it states the classification $class(\mathbf{x})$ of the nearest neighbor as the classification of the new case $(\mathbf{y}, ?)$, i.e. $(\mathbf{y}, class(\mathbf{x}))$. The basic algorithm (Aha, 1991) for a case-based approach is presented in Figure 1.

From the viewpoint of machine learning, case-based learning may be seen as a *concept formation task* (Richter, 1992). This raises the question how the learned concepts are represented in case-based approaches. Contrary to symbolic learning systems, which represent a learned concept *explicitly*, e.g. by formulas, rules, or decision trees, case-based systems describe a concept C *implicitly* (Holte, 1990) by a pair (CB, d). The relationship between the case base and the measure used for classification may be characterized by the equation:

$$\boxed{\textbf{Concept} = \textbf{Case Base} + \textbf{Distance Measure}}$$

In analogy to arithmetic this equation indicates that it is possible to represent a given concept C in multiple ways, i.e. there exist several pairs $C = (CB_1, d_1), (CB_2, d_2), \ldots, (CB_k, d_k)$ for the same concept C. Furthermore, the equation gives a hint how a case-based learner can improve its classification ability. There are in principle two possibilities to improve a case-based system. The system can

Basic Algorithm for Case-Based Classification

1. Define $CB = \{\ \}$ and initialize d
2. A new case $(\mathbf{y}, class(\mathbf{y}))$ is presented
3. Find a case $(\mathbf{x}, class(\mathbf{x})) \in CB$ so that $d(\mathbf{y}, \mathbf{x})$ is minimal.
4. If $class(\mathbf{y})$ is unknown, i.e. $(\mathbf{y}, ?)$ then
   (a) State $class(\mathbf{x})$ as classification of $(\mathbf{y}, ?)$, i.e. $(\mathbf{y}, class(\mathbf{x}))$.
   (b) Ask user for the correct classification $class(\mathbf{y})$ of $(\mathbf{y}, ?)$.
5. If $class(\mathbf{y}) = class(\mathbf{x})$
   then $classification := correct$
   else $classification := incorrect$
6. Modify d and/or CB with respect to $classification$.
7. Go to step 2

Figure 1: Basic Algorithm for a Case-Based Classifier

1. learn by changing the case base,
2. learn by changing the measure.

During the learning phase a case-based system (cf. Figure 1) gets a sequence of cases $X_1, X_2, \ldots, X_k$ with $X_i = (\mathbf{x}_i, class(\mathbf{x}_i))$ and computes a sequence of pairs (CB_1, d_1), $(CB_2, d_2), \ldots,$ (CB_k, d_k) with $CB_i \subseteq \{X_1, X_2, \ldots, X_i\}$. The aim is to get in the limit a pair (CB_n, d_n) that needs no further change, i.e. $\exists n\ \forall m \geq n\ (CB_n, d_n) = (CB_m, d_m)$, because it is a correct classifier for the target concept C.

2.2 Remarks

From the above description of the principle work of case-based and symbolic algorithms we can draw the following conclusions immediately.

- Both procedures produce a hypothesis when a new case is presented. Given only input and the classification behavior from the algorithms and the hypotheses, it is impossible to distinguish between the approaches.
- The hypotheses the algorithms produce work differently. The symbolic algorithm builds up its hypothesis by revealing the common characteristics of the examples in a pregiven hypothesis language. The hypothesis

describes the relation between an object and the concept. The main component of the hypothesis of a case-based learner is a measure that states the similarity or distance between objects. The measure defines a relation between two objects and is therefore independent from the existence of a concept.

- A main difference between case-based and symbolic algorithms is the representation of the learned concept. The hypothesis produced by a case-based algorithm represents the concept only *implicitly*, while symbolic procedures build up an *explicit* representation of the concept. It is often a non-trivial task to extract a symbolic representation of the concept from a case-base and a measure. Of course, in finite domains the extension of the concept can be determined by classifying all objects of the universe.
- If we abandon the modification of the solutions, we must assume that for all possible solutions a case will be presented and included in the case-base. Without the possibility to modify solutions the case-based learner is unable to produce new solutions.

Based on these characteristics a comparison of the algorithms must clarify the following questions (Globig, 1993).

- How can the hypotheses the different approaches produce be characterized and what is the relationship between the hypotheses of the approaches?
- Which class of problems is learnable by the algorithms? Are there differences in the learning or application phase?
- Are there hints when to prefer an approach?
- How does the algorithms solve typical problems?

The hypotheses the symbolic algorithms build up are predefined by the hypothesis language. We therefore confine ourselves to a characterization of the hypotheses of case-based learners.

3 Learning by changing the measure

We now transform a well-known symbolic learner – the Version Space (VS) from (Mitchell, 1982) – in a case-based variant. The Version Space algorithm is a simple and well-known symbolic learning algorithm. Because of its simplicity it is easy to show some properties that hold for many other learning algorithms, for which it would be difficult to prove them.

The case-based variant simulates the symbolic algorithm in the following sense. If an object is classified by the symbolic algorithm, then it is classified equally by the case-based variant.

3.1 Symbolic Version Space

The universe U of cases consists of finite vectors over finite value sets W_i ($U = W_1 \times \ldots \times W_n$). We want to decide the membership problem of a certain concept C. The concepts to learn fix the value of certain attributes.[2] These concepts C can be described as vectors $(C_1, \ldots, C_n)$, with $C_i = *$ or $C_i = a_{ij} \in W_i$. A case $((a_1, \ldots, a_n), class(\mathbf{a}))$ fulfills the concept C, if for all $1 \leq i \leq n$ holds: $C_i = *$ or $C_i = a_i$, i.e. $C_i = *$ is fulfilled by every $x \in W_i$. We further demand that $C_i \neq *$ for at least one i.

A concept C is called consistent with a set of cases, if all positive cases, i.e. $class(\mathbf{x}) = +$, of the set fulfill the concept and none of the negative, i.e. $class(\mathbf{x}) = -$, does. A concept C is called more general (more specific) than C' if $C \supset C'$ ($C \subset C'$). The symbolic version space (Mitchell, 1982) solves the learning problem by updating two sets S and G of concepts. S contains the most specific concept that is consistent with the known cases and G includes the most general concepts consistent with the known cases. The task of the symbolic algorithm is to change the sets S and G in order to preserve these properties. Figure 2 shows the algorithm (cf. (Mitchell, 1982)). For simplicity we assume that at first a positive case a^1 is given to initialize the sets.

It is important that at every moment all cases subsumed by S are known to be positive, and all cases that are not subsumed by any concept of G are known to be negative. If a case is presented that violates this condition, the target concept is not in the version space. This observation leads to a partial decision function $\mathrm{VS} : U \rightarrow \{0, 1\}$ that can be used to classify new cases:

$$\mathrm{VS}(\mathbf{x}) = \begin{cases} 1 & \text{if } \forall C \in S[C(\mathbf{x}) = 1] \\ 0 & \text{if } \forall C \in G[C(\mathbf{x}) = 0] \\ ? & \text{otherwise} \end{cases}$$

As long as $S \neq G$ holds VS will not classify all cases of the universe. If a case is covered by S but not by G it may belong to the concept C or not. So VS will not return an answer for those cases (this is the semantics of the "?" in the decision function).

3.1.1 Example

To illustrate this version space algorithm we present a very simple example. The universe U is $U = \text{shape} \times \text{size} = \{\text{Square, Circle}\} \times \{\text{big, small}\}$. Figure 3 shows the graph of all learnable concepts.

Let us study the changes of S and G during the learning process. If the first positive case is $((Circle, big), +)$ we have:

$$S = \{(Circle, big)\} \quad G = \{(*, *)\}$$

[2] These concepts represent the conjunctions of atomic formulas $x_i = a_i$, e.g. $shape = Circle \wedge size = big$.

Version Space Algorithm

1. Initialize $G = \{(*, \ldots, *)\}$ and $S = \{a^1\}$.

2. If the actual case is $(\mathbf{a}, +)$
   then remove all concepts from G that do not subsume the positive case.
   Search for the most specific concept C of the version space that subsumes all positive cases and define $S = \{C\}$. If there is no such C define $S = \emptyset$.

3. If the actual case is $(\mathbf{a}, -)$
   then remove all concepts from S which are fulfilled by $\mathbf{a}$.
   For all concepts $\mathbf{g} \in G$ that subsume $\mathbf{a}$, search for the most general specializations that do not subsume $\mathbf{a}$ but all known positive cases.
   Replace $\mathbf{g}$ by the found concepts.

4. If G or S is empty or there is a concept $\mathbf{g}$ in G that is more specific than the concept from S, then **ERROR:** Not a concept of the version space!

5. If $S = G$ then **STOP:** Concept $= S$
   else go to 2.

Figure 2: Algorithm for the symbolic version space

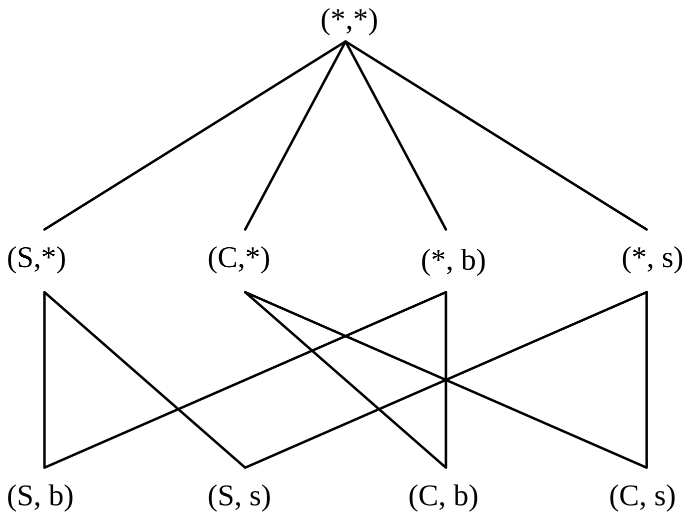

Figure 3: Set of the learnable concepts over U

Let the second case be negative $((Square, small), -)$. This forces the algorithm to specialize the concept in G. Because all concepts that replace $(*,*)$ must be consistent with the known cases, the most general specialization are $(*, big), (Circle, *)$. So, S and G change to:

$$S = \{(Circle, big)\} \quad G = \{(*, big), (Circle, *)\}$$

If the third case $((Square, big), +)$ is positive we must generalize the concept in S and specialize the concept in G. The only possible concept consistent with the cases is $(*, big)$. S and G turn out to be:

$$S = \{(*, big)\} \quad G = \{(*, big)\}$$

Now S and G are equal and contain only a single concept. The learned concept $C = (*, big)$ is defined totally, i.e. for every case of the universe it is possible to decide whether it fulfills the target concept. If we present more cases, the sets S and G will not change.

3.2 Case-Based Version Space

If we analyze the version space algorithm, it is obvious that the main learning task is to distinguish between relevant and irrelevant attributes. We will use this observation to construct a case-based variant VS-CBR of the algorithm of the previous section. An attribute value is called *relevant*, if it is part of the target concept $\mathbf{C} = (C_1, \ldots, C_n)$ ($C_i \in W_i \cup \{*\}$). For every attribute i, we define a function f_i that maps $x \in W_i$ to $\{0,1\}$ with the following definition:

$$f_i(x) = \begin{cases} 1 & : \quad C_i = x \\ 0 & : \quad \text{otherwise} \end{cases}$$

Note that $f_i \equiv 0$ if $C_i = *$. That means the value of the ith attribute does not influence the measure of similarity. The functions f_i will be combined to $\mathbf{f}\colon U \to \{0,1\}^n$ $\mathbf{f}((a_1, \ldots, a_n)) = (f_1(a_1), \ldots, f_n(a_n))$. The distance between two cases a and b is then defined using the city-block metric as follows

$$d_{\mathbf{f}}(\mathbf{a}, \mathbf{b}) := |f_1(a_1) - f_1(b_1)| + \ldots + |f_n(a_n) - f_n(b_n)|$$

It is obvious that every change of the functions $f_1, f_2, \ldots, f_n$ causes a change of the underlying measure $d_{\mathbf{f}}$. The intended function f_i is learnable by the algorithm in Figure 4. The algorithm expects the first case to be positive.[3]

Algorithm to Learn f for VS-CBR

1. Initialize $f_i(x_i) = 0$ for all $i, x_i \in W_i$.
2. Let the first positive case be $((a_1, \ldots, a_n), +)$. Set $f_i(a_i) = 1$ for all i and $CB = \{(\mathbf{a}, +)\}$.
3. Get a new case $((b_1, \ldots, b_n), class(\mathbf{b}))$.
4. If $class(\mathbf{b})$ is negative, store $\mathbf{b}$ in the case base CB, i.e. $CB := CB \cup \{(\mathbf{b}, -)\}$.
5. If $class(\mathbf{b})$ is positive and $f_i(b_i) = 0$, then let $f_i(x_i) = 0$ for all $x_i \in W_i$ (f_i maps now every value to zero).
6. If there exist two cases $(\mathbf{a}, class(\mathbf{a})), (\mathbf{b}, class(\mathbf{b})) \in CB$ with $d_{\mathbf{f}}(\mathbf{a}, \mathbf{b}) = 0$ and $class(\mathbf{a}) \neq class(\mathbf{b})$ then **ERROR:** The target concept C is not member of the version space.
7. If the concept C is determined then **STOP:** The concept is learned. The classifier $(CB, d_{\mathbf{f}})$ consists of the case base CB and the measure $d_{\mathbf{f}}$.
8. Go to step 3.

Figure 4: Algorithm to learn **f** for VS-CBR

If the concept is learned, the function **f** and the case base CB are used for

[3] If the first case is not positive, the learner may store all negative cases and start the algorithm, if the first positive case is presented.

classification. Given a new case $(c, ?)$, the set

$$F := \{\mathbf{x} \in CB \mid \forall\ y\ \in CB\ d_{\mathbf{f}}(\mathbf{x}, c) \leq d_{\mathbf{f}}(y, c)\}$$

is computed. The classification $class(\mathbf{x})$ of the nearest neighbor $(\mathbf{x},\ class(\mathbf{x}))$ is then used for the classification of the new case $(\mathbf{c}, ?)$. If F contains more than one case with different classifications then $class(\mathbf{c})$ is determined by a fixed strategy to solve this conflict. Different strategies are possible and each strategy will induce its own semantics for VS-CBR.

For example, one conflict solving strategy may state the minimal classification according to a given ordering of the concepts. This strategy is used in the following decision function:

$$\text{VS-CBR}(\mathbf{x}) = \min\{class(\mathbf{y}) \mid \mathbf{y} \in CB\ \wedge\ \forall\ \mathbf{z} \in CB\ d_{\mathbf{f}}(\mathbf{y}, \mathbf{x}) \leq d_{\mathbf{f}}(\mathbf{z}, \mathbf{x})\}$$

To solve the membership problem, we assume that a case $(\mathbf{c}, ?)$ is classified as negative if it has the same minimal distance from a positive and a negative case, i.e. $d((\mathbf{a}, +), (\mathbf{c}, ?)) = d((\mathbf{b}, -), (\mathbf{c}, ?))$ is minimal. To achieve this behavior of the classifier the ordering of the concepts must be $"-" < "+"$.

3.2.1 Example

Before analyzing the classification ability of VS-CBR in more detail, we illustrate the algorithm by the same simple example we have used for the VS (cf. section 3.1.1). Because the universe has only two dimensions, two functions $f_1 : shape \rightarrow \{0, 1\}$ and $f_2 : size \rightarrow \{0, 1\}$ are needed. The first positive case $((Circle, big), +)$ is used to initialize the functions f_1 and f_2.

$$\begin{aligned} f_1(x) &= \begin{cases} 1 & \text{if } x = Circle \\ 0 & \text{otherwise} \end{cases} \\ f_2(y) &= \begin{cases} 1 & \text{if } y = big \\ 0 & \text{otherwise} \end{cases} \\ CB &= \{((Circle, big), +)\} \end{aligned}$$

The next case of our sequence is $((Square, small), -)$. This new case is stored in the case base.

$$\begin{aligned} f_1(x) &= \begin{cases} 1 & \text{if } x = Circle \\ 0 & \text{otherwise} \end{cases} \\ f_2(y) &= \begin{cases} 1 & \text{if } y = big \\ 0 & \text{otherwise} \end{cases} \\ CB &= \{((Circle, big), +), ((Square, small), -)\} \end{aligned}$$

Now $(CB, d_{\mathbf{f}})$ classifies $((Circle, big), +)$ as positive only, because every other case has a distance ≥ 1 from $((Circle, big), +)$ and ≤ 1 from $((Square, small), -)$.

As third case assume $((Square, big), +)$. Because $f_1(Square) = 0$ holds f_1 is defined to be zero for all values. The new case is not stored in the case base.

$$
\begin{aligned}
f_1(x) &= 0 \\
f_2(y) &= \begin{cases} 1 & \text{if } y = big \\ 0 & \text{otherwise} \end{cases} \\
CB &= \{((Circle, big), +), ((Square, small), -)\}
\end{aligned}
$$

We may now test the elements of the universe U. They are all correctly classified. However, it is not obvious from the algorithm why the learning process can be stopped at this point.

3.3 Analysis

Now let us analyze VS-CBR's way of classification in more detail. Positive and negative cases are used differently in VS-CBR during the learning phase:

- Positive cases are used to change $\mathbf{f}$, i.e. to adapt the distance measure $d_{\mathbf{f}}$. They will not be stored in the case base (with the exception of the very first positive case).
- Negative cases are stored in the case base CB but do not change the distance measure.

The information that is used by VS to change S and G is used by VS-CBR to change the case base or the distance measure.

It is easy to show that all cases which are classified by the symbolic VS will also be classified correctly by the case-based one. The difference is that the case-based variant VS-CBR computes a classification for every case of the universe (because the distance measure is total) while the symbolic VS classifies only if the proposed classification can be proven to be correct. Otherwise (i.e. the case fulfills a concept from G but not the concept in S) it will not produce any classification at all. If we add a test whether the classification of the nearest neighbor is correct to VS-CBR, we can force VS-CBR to produce only certain classifications, too. But this test would more or less be a variant of the original VS algorithm.

We have shown that it is possible to reformulate the Version Space algorithm in a case-based manner so that the case-based variant simulates the symbolic algorithm. As we have seen a case-based learning system consists of two main parts: the case base and the distance measure. Therefore, we want to analyze the implications of the choice of the distance measure and the strategy to select cases for the case base.

3.4 Comparing different measures

In this section consequences of the choice of d are drawn. For the rest of section 3 we assume the following scenario:

1. The universe U of cases is finite.
2. We have to decide the membership problem of a certain concept C.
3. The distance measure d is total and satisfies the following condition: $\forall \mathbf{a}, \mathbf{b}, \mathbf{x} \in U\, [d(\mathbf{a}, \mathbf{a}) = 0 \wedge (d(\mathbf{a}, \mathbf{b}) = 0 \Rightarrow d(\mathbf{x}, \mathbf{a}) = d(\mathbf{x}, \mathbf{b}))]$.

Condition 3 has two important consequences: First, the relation $\sim \subseteq U \times U$ defined by $\mathbf{x} \sim \mathbf{y} \Leftrightarrow d(\mathbf{x}, \mathbf{y}) = 0$ is an equivalence relation. Second, all members of the equivalence relation must have the same classification because there cannot exist any case to separate them. $|U/\sim|$ is the number of equivalence classes that are induced by $\sim$. So we can state that d is able to represent exactly those concepts C that satisfy $d(\mathbf{x}, \mathbf{y}) = 0 \Rightarrow C(\mathbf{x}) \equiv C(\mathbf{y})$, i.e. the members of an equivalence class must have the same classification.

The measure d is able to distinguish between $2^{|U/\sim|}$ different concepts C_j. Each concept can be represented by almost $|U/\sim|$ (appropriate) cases. In other words, in a case-based classifier (CB, d) the measure d defines the set of the learnable concepts and the case base CB selects a concept from this set.

During the learning process the case-based system alters. On one hand, case-based systems (CB, d) use the cases in the case base CB to fill up the equivalence classes induced by the measure d. On the other hand, they use the cases to lower the number of equivalence classes by changing the measure d. Thereby, the target concept C may be identified by fewer cases. But, a lower number of equivalence classes means that the modified measure d' can distinguish between fewer concepts.

Having this in mind, we can compare case-based systems with respect to two dimensions: *minimality* and *universality*. The first dimension relates to the implicit knowledge that is coded into the used measure d. Because we are not able to measure this implicit knowledge directly, we have to look at the size of the case base instead. More knowledge coded in the used measure d will result in a smaller (minimal) size of the case base CB within the classifier (CB, d).

Definition 1 *The measure d_1 is called* better informed *about a concept* $\mathbf{C}$ *than a measure d_2 if*[4]

$$\exists CB_1 \subset_{fin} U \quad \forall CB_2 \subset_{fin} U \quad [(CB_1, d_1) = \mathbf{C} = (CB_2, d_2) \Rightarrow |CB_1| \leq |CB_2|]$$

[4] $A \subset_{fin} B$ denotes that A is a finite subset of B

The second dimension relates to the set of representable concepts. We must distinguish between the representability and the learnability of a concept. A concept C is called representable by a measure d, if there *exists* a finite case base CB such that (CB, d) is a classifier for C. A concept C is called learnable by a measure d, if there exists a *strategy to build* a finite case base CB such that in the limit (CB, d) is a classifier for the concept.

Definition 2 *A measure d_1 is called* more universal *than a measure d_2 iff the set of concepts that are representable by d_2 is a proper subset of the set of concepts that are representable by d_1.*

Using an universal measure conflicts the minimality of the case base. Reducing the size of the case base, which means to code more knowledge into the measure, usually results in a less universal measure. We can distinguish two extreme situations:

All knowledge is coded into the case base: The measure is minimal if and only if the compared cases are identical, i.e.

$$d_{CB}(\mathbf{x}, \mathbf{y}) := \begin{cases} 1 & \text{if } \mathbf{x} \neq \mathbf{y} \\ 0 & \text{if otherwise} \end{cases}$$

The measure d_{CB} is universal because it is able to learn every binary concept C_i in the given universe U. But to do so, it needs the whole universe as a case base, i.e. $CB := U$. Thus, the resulting system (U, d_{CB}) is universal but not minimal.

All knowledge is coded into the measure: The measure is minimal if and only if the classification of the compared cases is identical, i.e. the measure d_C knows the definition of the target concept C.

$$d_C(\mathbf{x}, \mathbf{y}) := \begin{cases} 1 & \text{if } (C(\mathbf{x}) \neq C(\mathbf{y})) \\ 0 & \text{if otherwise} \end{cases}$$

Nearly the whole knowledge about the concept is then coded into the measure d_C. The case base contains almost one positive case $\mathbf{c}^+$ and one negative case $\mathbf{c}^-$ and is used only to choose between some trivial variations. The measure d_C can only distinguish between four concepts (C, $\neg C$, $True$ – i.e. all cases are positive, $False$ – i.e. all cases are negative). Thus, the resulting system $(\{\mathbf{c}^+, \mathbf{c}^-\}, d_C)$ is minimal but not universal.

We illustrate the contrasting nature of these two aims in Figure 5 by an example. This figure shows different measures d in relation to the minimal size of the case base CB to learn a certain concept C in the relation to the total number of learnable concepts. For the table, we use a universe U of cases that consist of four attributes. Each attribute can take one value out of 16. So, the

Used measure	Minimal size of CB	# Represent. concepts
d_{CB}	$65536 = 16^4$	2^{65536}
d_f^1	16	$65536 = 2^{16}$
d_f^n	4	$16 = 2^4$
d_C	2	2^2

Figure 5: Comparing different measures

size of the universe U is 65536. The concept the measures try to learn, fix two attributes out of four.

The universal measure d_{CB} is able to represent all binary concepts, while the minimal measure d_C needs only two cases to represent the learned concept. The other measures in the Figure 5 are between these two extremes. The measures d_f^1 and d_f^n are neither maximally universal nor able to represent the concept with a minimal case base. d_f^1 is the distance measure computed for VS–CBR after the first case has been presented. In every dimension exactly one value is mapped to 1. The universe U is therefore mapped onto the vertices of a four dimensional cube. d_f^n is the measure used when VS–CBR has learned the concept. It distinguishes only between the two relevant values of the concept and consequently builds up only four equivalence classes.

We can draw the following conclusions from these observations.

- Changing the used measure by coding more knowledge into it means trading universality against minimality.
- In a case-based learner, two processes – reducing the set of the representable concepts (hypothesis space) and increasing the size of the case base – should be performed.
- The last measure in Figure 5 indicates a simple way to reformulate any symbolic algorithm in a case-based manner, i.e. use the actual symbolic hypothesis to construct such a measure and store one positive and one negative case in the case base.

3.5 Using extended measures

We have shown that under the assumptions of this section a concept C is representable if and only if $d(\mathbf{x}, \mathbf{y}) = 0 \Rightarrow C(\mathbf{x}) \equiv C(\mathbf{y})$. Whether a concept C is representable by a given distance measure d, therefore, depends on the definition of the identity, i.e. the distance $d(\mathbf{x}, \mathbf{y}) = 0$, only. If the concept C

Figure 6: Graphical representation of concept C (cf. page 16)

is representable by d, all other distances may be mapped to any value greater than zero. This poses the question if it does make any sense to use a distance measure d that maps distances between cases to a greater set of values than $\{0, 1\}$.

The only reason to use a more complex distance measure (in our simple scenario; cf. (Wess, 1993)) is the hope to get more reliable hypotheses before the concept C is learned, i.e. when not all equivalence classes of the measure are filled.

3.5.1 Example

To illustrate this, we compare two measures that can learn the same concepts. Let the universe consist of cases with four attributes. The values for each attribute are $\{0, \ldots, 15\}$. So the universe is $U = \{0, \ldots, 15\}^4$. The concept to learn is $C(\mathbf{x})$, $\mathbf{x} := (x_1, \ldots, x_4)$ (cf. Figure 6).

$$C(\mathbf{x}) = 1 \Leftrightarrow (x_2 \geq 8 \wedge x_4 < 8) \vee (x_4 \geq 8 \wedge x_2 < 8)$$

and the distance measures are:

$$\begin{aligned} d_1(\mathbf{a}, \mathbf{b}) &= \begin{cases} 0 & : \mathbf{a} = \mathbf{b} \\ 1 & : \text{otherwise} \end{cases} \\ d_2(\mathbf{a}, \mathbf{b}) &= \sum_i |a_i - b_i| \end{aligned}$$

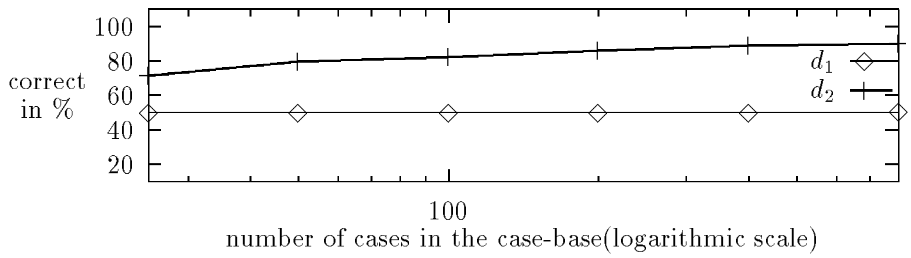

Figure 7: Classification rate in percent for d_1 and d_2 with cases bases of different size

Obviously both measures are able to represent the same concepts in the universe U, because $d_i(\mathbf{a}, \mathbf{b}) = 0 \Leftrightarrow \mathbf{a} = \mathbf{b}$. Figure 7 shows the classification rates for the given concept $C(\mathbf{x})$ with case bases of different sizes. To classify more than 90% correctly with the measure d_1 more than 52000 cases, are required.

The portion of correctly classified cases grows with the measure d_2 while it is nearly constant with measure d_1, i.e. d_2 of (CB_2, d_2) is a much better informed measure than the d_1 of (CB_1, d_1). The difference between the measures d_1 and d_2 is the significance of the distance value. If d_1 measures a distance greater than 0 there is no hint whether the classification of the cases is identical or not. On the other hand, a small distance measured by d_2 indicates a high probability that the cases can be equally classified.

This result does not imply that d_2 is the best choice for all concepts (Globig & Wess, 1994). It is possible to define concepts where a small distance between cases implies a high probability for different classification. An example is the following concept (cf. Figure 8):

$$\begin{aligned} C_2(\mathbf{x}) = 1 \quad \Leftrightarrow \quad & ((x_2 \mod 2 = 0) \wedge (x_4 \mod 2 = 1)) \vee \\ & ((x_4 \mod 2 = 0) \wedge (x_2 \mod 2 = 1)) \end{aligned}$$

The rate of correct classifications for d_2 and concept C_2 will be nearly the same as the rate of d_1 in Figure 7.

4 Learning by changing the case base

In the last section we analyzed some aspects of the choice of the distance measure. In this section we want discuss the influence of the choice of the case

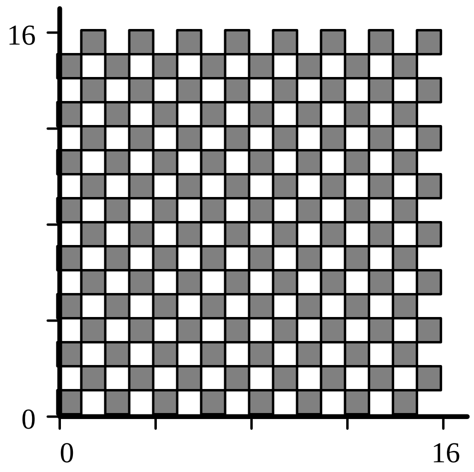

Figure 8: Graphical representation of concept C_2 (cf. page 17)

selection strategy on the set of the learnable concepts. The example systems of the last section store all presented cases. In this section the measure of similarity is fixed. In (Jantke, 1992) Jantke has already shown that a case-based system can simulate an inductive learning algorithm if the similarity measure can be changed arbitrarily.

We have done this comparisons in the area of learning indexed families of formal languages (cf. (Angluin, 1980)). This comparison was done by one of the authors together with Steffen Lange. A more detailed description of theses results including the proofs of the theorems can be found in (Globig & Lange, 1994).

The definitions of this section are adapted from the Inductive Inference literature (cf. (Angluin, 1980)). Our target objects are (formal) languages over a finite alphabet A. By A^+ we denote the set of all non-empty strings over the alphabet A. Any subset L of A^+ is called a language. We set $\overline{L} = A^+ \setminus L$.

By $\mathbb{N} = \{1, 2, \ldots\}$ we denote the set of all natural numbers. We use $\mathbb{Q}_{[0,1]}$ to denote the set of all rational numbers between 0 and 1. Furthermore, by $card(B)$ we denote the cardinality of set B. We write $B \# C$ if B, and C are incomparable with respect to inclusion.

There are two basic ways to present information about a language to a learner. We can present positive data only or positive and negative data. These presentations are called *text* and *informant*, respectively. A *text* for a language L is an infinite sequence $t = (s_1, b_1), (s_2, b_2), \ldots$ with $(s_j, b_j) \in A^+ \times \{+\}$ such that $\{s_j \mid j \in \mathbb{N}\} = L$. Let $t[k]$ be the initial sequence $(s_1, b_1), (s_2, b_2), \ldots, (s_k, b_k)$ of t. We set $t^+[k] = \{s_j \mid j \leq k\}$. Let $text(L)$ denote the set of all texts of L. An *informant* for a language L is an infinite sequence $i = (s_1, b_1), (s_2, b_2), \ldots$ with $(s_j, b_j) \in A^+ \times \{+, -\}$ such that $\{s_j \mid j \in \mathbb{N},\ b_j = +\} = L$ and $\{s_j \mid j \in \mathbb{N},\ b_j = -\} = A^+ \backslash L$. Let $i[k]$ be the initial sequence $(s_1, b_1), (s_2, b_2), \ldots, (s_k, b_k)$ of i. Furthermore, we set $i^+[k] = \{s_j \mid j \leq k,\ b_j = +\}$ and $i^-[k] = \{s_j \mid j \leq k,\ b_j = -\}$. By *informant*$(L)$ we denote the set of all informants of L. Without loss of generality we assume that $t[k]$ ($i[k]$) is coded as a natural number that represents the initial segment of the text (resp. informant).

We restrict ourselves to investigate the learnability of indexed families of recursive languages over A (cf. (Angluin, 1980)). A sequence $\mathcal{L} = L_1, L_2, \ldots$ is said to be an *indexed family* if all L_j are non-empty and there is a recursive function f such that for all indices j and all strings $w \in A^+$ holds

$$f(j, w) = \begin{cases} 1 & \text{if } w \in L_j \\ 0 & \text{otherwise} \end{cases}$$

So given an indexed family $\mathcal{L}$ the membership problem is uniformly decidable for all languages in $\mathcal{L}$ by a single function.

IF denotes the set of all indexed families.

The following definition is adapted from (Angluin, 1980). We use $f(x)\downarrow$ to denote that a function f is defined on input x.

Definition 3 *Let* $\mathcal{L} \in \mathbf{IF}$.
Then we say $\mathcal{L}$ *is* learnable from text *(resp.* learnable from informant*)*
iff
$\exists M \in \mathbf{P}\ \forall L \in \mathcal{L}\ \forall t \in$ *text*(L) *(resp.* $\forall i \in$ *informant*(L)*)*

$$\begin{array}{ll} (1) & \forall n \in \mathbb{N}\ M(t[n])\downarrow\ (\textit{resp. } \forall n \in \mathbb{N}\ M(i[n])\downarrow), \\ (2) & \lim_{n\to\infty} M(t[n]) = a \textit{ exists } (\textit{resp. } \lim_{n\to\infty} M(i[n]) = a \textit{ exists}), \\ (3) & L_a = L. \end{array}$$

LIM.TXT (**LIM.INF**) *is the set of all indexed families that are learnable from text (informant).*

P denotes the set of the unary computable functions.

In order to formalize case-based learnability we have to define the underlying similarity measures. $\sigma : A^+ \times A^+ \to \mathbb{Q}_{[0,1]}$ is called a measure of similarity. Σ denotes the set of all totally defined and computable similarity measures.

To define case-based learnability in this setting, we use the so called standard semantics L_{st} (cf. (Jantke & Lange, 1993)).

Definition 4 *Let* $CB \subseteq_{fin} A^+ \times \{+, -\}$ *and* $\sigma \in \Sigma$ *a similarity measure. Furthermore, let* $CB^+ := \{s \mid (s, +) \in CB\}$, $CB^- := \{s \mid (s, -) \in CB\}$. *Then we say* CB *and* σ *describe the language* $L_{st}(CB, \sigma) = L_{st}(CB^+, CB^-, \sigma) := \{w \in A^+ \mid \exists c \in CB^+\ (\sigma(c, w) > 0 \wedge \forall c' \in CB^- \sigma(c, w) > \sigma(c', w))\}$.

Definition 5 *Let* $\mathcal{L} \in \mathbf{IF}$ *and* $\sigma \in \Sigma$.
Then, $\mathcal{L} \in \mathbf{REPR}^+(\sigma)$ *iff for every* $L \in \mathcal{L}$ *there is a* $CB^+ \subseteq_{fin} L$ *such that* $L_{st}(CB^+, \emptyset, \sigma) = L$. *Moreover,* $\mathcal{L} \in \mathbf{REPR}^{\pm}(\sigma)$ *iff for every* $L \in \mathcal{L}$ *there are* $CB^+ \subseteq_{fin} L$ *and* $CB^- \subseteq_{fin} \overline{L}$ *such that* $L_{st}(CB^+, CB^-, \sigma) = L$.
Let $\mathbf{REPR}^+ := \bigcup_{\sigma \in \Sigma} \mathbf{REPR}^+(\sigma)$ *and* $\mathbf{REPR}^{\pm} := \bigcup_{\sigma \in \Sigma} \mathbf{REPR}^{\pm}(\sigma)$.

So $\mathcal{L} \in \mathbf{REPR}^+$ ($\mathcal{L} \in \mathbf{REPR}^{\pm}$) means that there is a σ such that $\mathcal{L} \in \mathbf{REPR}^+(\sigma)$ ($\mathcal{L} \in \mathbf{REPR}^{\pm}(\sigma)$).

Based on the classical definitions we define case-based learnability with respect to a certain case selection strategy.

Definition 6 *An indexed family* $\mathcal{L}$ *is said to be* case-based learnable from text *by the case selection strategy* $S : \mathbb{N} \to Pot(A^+ \times \{+\})$ *iff*
$\exists \sigma \in \Sigma \ \forall L \in \mathcal{L} \ \forall t \in \ text(L)$

(1) $\forall n \in \mathbb{N} \ CB_n = S(t[n]) \downarrow$, *and* $S(t[n]) \subseteq t^+[n] \times \{+\}$,

(2) $CB = \lim_{n \to \infty} CB_n$ *exists and* CB *is finite,*

(3) $L_{st}(CB, \sigma) = L$.

Definition 7 *An indexed family* $\mathcal{L}$ *is said to be* case-based learnable from informant *by the case selection strategy* $S : \mathbb{N} \to Pot(A^+ \times \{+, -\})$ *iff*
$\exists \sigma \in \Sigma \ \forall L \in \mathcal{L} \ \forall i \in \ informant(L)$

(1) $\forall n \in \mathbb{N} \ CB_n = S(i[n]) \downarrow$, *and* $S(i[n]) \subseteq (i^+[n] \times \{+\}) \cup (i^-[n] \times \{-\})$,

(2) $CB = \lim_{n \to \infty} CB_n$ *exists and* CB *is finite,*

(3) $L_{st}(CB, \sigma) = L$.

Here the learner is not allowed to change the measure of similarity during the learning process. Therefore, the learning capability depends on the case selection strategy only.

In the sequel we want to analyze the influence of two dimensions – access to case history and deleting cases form the case base.

Access to case history: Is the case selection strategy allowed to store any case that is already presented or has the strategy access to the last one only?

Deleting cases from the case base: Is the case selection strategy allowed to delete cases from the case base or does the case base grow monotonically?

With respect to these dimensions we can define types of case selection strategies. Let CB_k be the case base constructed when a learner has seen an initial sequence of length k.

Definition 8 *Let S be a case selection strategy. Then S is said to be of type*[5] **MO-LC**, **MO-RA**, **DE-LC**, *and* **DE-RA**, *respectively, iff the corresponding condition holds for all $k \in \mathbb{N}$ ($CB_0 := \emptyset$).*

MO-LC	*iff* $CB_{k-1} \subseteq CB_k \subseteq CB_{k-1} \cup \{(s_k, b_k)\}$
MO-RA	*iff* $CB_{k-1} \subseteq CB_k \subseteq \{(s_1, b_1), \ldots, (s_k, b_k)\}$
DE-LC	*iff* $CB_k \subseteq CB_{k-1} \cup \{(s_k, b_k)\}$
DE-RA	*iff* $CB_k \subseteq \{(s_1, b_1), \ldots, (s_k, b_k)\}$

We use these abbreviations as prefixes to **CBL.TXT** and **CBL.INF**. For example, $\mathcal{L} \in$ **DE-RA-CBL.TXT** means that there is a case selection strategy $S \in$ **DE-RA** such that $\mathcal{L}$ can be learned by S in the sense of Definition 6.

Strategies of type **MO-RA** and **DE-RA**, respectively, may store multiple cases in a single learning step. If we demand that strategies of both types store at most a single case in every learning step their learning capabilities will not change.

Because many existing systems simply collect all presented cases, we model this approach, too. A case selection strategy S is said to be of type[6] **CA**, if $CB_k = \{(s_j, b_j) \mid j \leq k\}$ for all $k \in \mathbb{N}$.

It is possible that a **CA-CBL.TXT**-strategy leads to a case base of infinite size, for instance, if the language that is described by a text is infinite. So we have to define what it means that such a strategy learns successfully.

Definition 9 *Let $\mathcal{L}$ be an indexed family. We say $\mathcal{L} \in$* **CA-CBL.TXT** *iff* $\exists \sigma \in \Sigma\ \forall L \in \mathcal{L}\ \forall t \in\ text(L)$

$$\begin{array}{ll} (1) & \forall n \in \mathbb{N}\ CB_n = t^+[n] \times \{+\}, \\ (2) & \exists j \in \mathbb{N}\ L_{st}(CB_k, \sigma) = L \textit{ for all } k > j. \end{array}$$

CA-CBL.INF *is defined analogously.*

We say $\mathcal{L} \in$ **CA-CBL.TXT** if for all texts of L, $(L_{st}(CB_n, \sigma))_{n \in \mathbb{N}}$ converges *semantically*. This is somehow comparable to the notion of convergence underlying the identification type **BC** in Inductive Inference of recursive functions (Angluin & Smith, 1983). All other case-based learning types demand that the sequence $(CB_n)_{n \in \mathbb{N}}$ itself has to converge.

4.1 Learning from Text

The two main results concerning case-based learning from text are contained in the following theorem (cf. (Globig & Lange, 1994)).

[5] *MO* stands for "monotonically", *DE* for "delete", *RA* for "random access" and *LC* for "last case"

[6] *CA* stands for "collect all"

Theorem 1
LIM.TXT # REPR$^+$
LIM.TXT $\cap$ REPR$^+$ = DE-RA-CBL.TXT

The first part says, that neither all representable indexed families are learnable nor all learnable families representable. So the lack of learning power of case-based learning from text is due to the lack of representability of indexed families with positive cases. Problems arise when the indexed family contains both finite and infinite languages. Indexed Families that are representable with positive cases only are learnable form text with the most flexible case selection strategy.

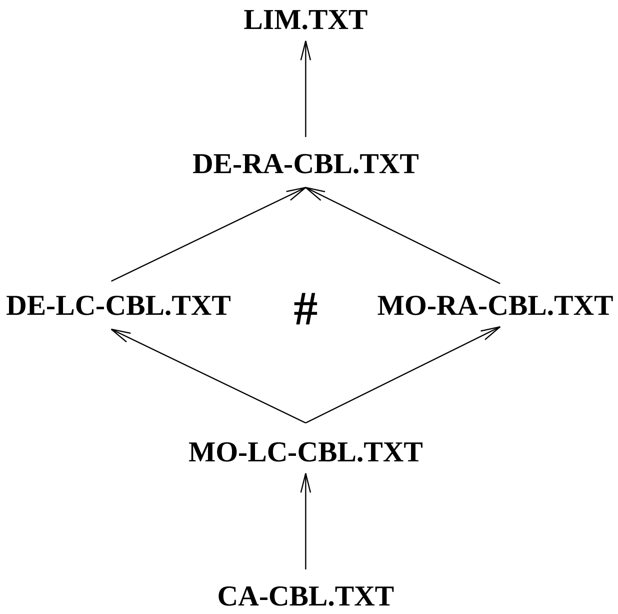

Figure 9: Relationships between the learning types

Figure 9 shows the relationships between the defined case-based learning types. If there is a path from T_1 to T_2, T_1 is a proper subset of T_2. The figure indicates that both random access to the already presented cases and the ability to delete cases from the actual case base increase the learning power of a case-based learning system. But neither subsumes the other. Note that even if we allow random access to the presented cases and deleting cases from the case base the full learning power is not reached.

Collecting all cases results not only in a slow system (because of the size of the case base) but also reduces the learning power. This reveals the power of selecting appropriate cases. The more flexible the case selection strategy is, the more classes can be learned.

4.2 Learning from Informant

Learning form informant is more powerful than learning from text. It is well known that every indexed family is learnable from informant. The following theorems show that this result is valid for case-based learnability, too.

Theorem 2
(1) **MO-LC-CBL.INF** $\subset$ **IF**
(2) **MO-RA-CBL.INF** = **IF**
(3) **DE-LC-CBL.INF** = **IF**

Corollary 1 **DE-RA-CBL.INF** = **IF**

It is not only remarkable that all indexed families are case-based learnable but by comparatively simple selection strategies. While in the text-case even the most flexible case selection strategies were unable to learn all classes in **LIM-TXT** in the informant-case random access or deleting from the case base are enough.

The proof of third equation of Theorem 2 is based on a similarity measure that allows to represent every language of an indexed family using at most two cases.

5 Discussion

The symbolic as well as the case-based approach compute a classification when a new case is presented. If only the input and the output of the algorithms are known, we will not be able to distinguish between the symbolic and the case-based approach. The symbolic algorithm builds up its hypothesis by revealing the *common characteristics* of the cases in a predefined *hypothesis language*. The hypothesis describes the *relation between a case and the concept*. One component of a case-based learner is a measure that states the similarity or the distance between cases.

A main difference between case-based and symbolic classification algorithms is the representation of the learned concept (cf. section 2.1). A case-based classifier (CB, d) consists of a case base CB and a measure of similarity d. It is possible to represent the same concept C in multiple ways, i.e. by different tuples (CB_i, d_i). But, neither the case base CB nor the measure sim is sufficient to build a classifier for C. The knowledge about the concept C is spread to both. Thus, the hypothesis produced by a case-based algorithm represents the concept only *implicitly*, while symbolic procedures build up an *explicit* representation of the learned concept. Often it is a non-trivial task to extract an explicit symbolic representation of the concept from a case base and a measure.

We have shown a method (cf. section 3.4) to reformulate a symbolic learning approach into an equivalent case-based variant. If the problems and the power of case-based and symbolic approaches are similar (Jantke, 1992) as we have seen for our simple scenario (cf. section 3.2), the question arises whether the two approaches can be interchanged in all situations. We assume that we want to get a classifier only and not an explicit description of the concept. In the second case, a case-based system cannot be the appropriate choice. Within this perspective, the symbolic and the case-based approach seem to be interchangeable in the described context. The symbolic approach corresponds to a kind of *compilation process* whereas the case-based approach can be seen as a kind of *interpretation* during run time. Which approach should be used in a concrete situation is a question of an adequate *representation of the previous knowledge*. If previous knowledge contains a *concept of neighborhood* that leads to appropriate hypotheses (like in section 3.5.1), a case-based approach is a good choice. In this scenario we are able to code the neighborhood principle into the used measure. The case-based approach will then produce good hypotheses before the concept is learned, i.e. when not all equivalence classes of the measure are filled.

We have analyzed (cf. section 3.4) the relationship between the measure of similarity, the case base, and the target concept in the described scenario of classification tasks (Globig, 1993). The learning algorithm *needs strong assumptions* about the target concept in order to solve its task with an acceptable number of cases. Assumptions exclude certain concepts from the hypothesis space. Symbolic learners use these assumptions to restrict the language to represent their hypotheses. A case-based learner has to code this assumptions into the measure of similarity. These restrictions of the hypothesis space are called *bias*. (Rendell, 1986) divides the abstraction done by a learning system in two parts: *the bias* (to describe the amount of assumptions), and the *power of the learner*. We have characterized (cf. section 3.4) case-based systems by the *number of learnable concepts* and the *number of cases* they need to identify a target concept. Case-based algorithms use the cases of the case base to fill equivalence classes induced by the measure used. On the other hand, they use the knowledge from the cases to lower the number of equivalence classes by changing the measure. Thereby, the target concept may be identified by fewer cases. The used measure defines the set of the learnable concepts and the cases in the case base select a concept from this set. The *bias* relates to the restriction of the set of learnable concepts induced by the measure of similarity and is therefore comparable to the *degree of universality*. The *minimal size* of the case base reflects the information the learner needs to come to a correct hypothesis, i.e. the power of the learner (Rendell, 1986). Using an universal similarity measure conflicts the minimality of the case base. Reducing the size of the case base, which means to code more knowledge into the measure, usually results in a less universal similarity measure. In section 4 we have seen that if we use a fixed measure of similarity we need advanced case selection strategies to improve the learning power. The knowledge how to select the appropriate cases is of course knowledge about the class of target concepts.

We have stressed that the measure (respectively the way to modify the measure) is the *bias of case-based reasoning*. Without any bias inductive learning is impossible with an acceptable amount of time. Without restrictions of the hypothesis space, neither symbolic nor case-based systems are able to learn even in a finite universe. Because case-based systems are based on a bias that cannot be deduced from the cases, we reject the thesis that case-based classification is more appropriate in situations with a low amount of previous knowledge.

In the last section we studied different types of case-based learning of indexed families from positive data and both positive and negative data with respect to an arbitrary fixed similarity measure. Thereby, we focused our attention on the problem of how the underlying case selection strategies influence the capabilities of case-based learners. As it turns out, the choice of the case selection strategy is of particular importance, if case-based learning from text is investigated. If both positive and negative data are provided, even quite simple case selection strategies are sufficient in order to exhaust the full power of case-based learning.

We conclude that for classification tasks there is no fundamental advantage in the learning power of case-based systems. As we have seen (cf. section 3.5.1) the intelligibleness of the classifications of a case-based system depends on the intelligibleness of the measure of similarity and is therefore not a property of the case-based approach itself. Since the number of cases an algorithm needs to learn a concept is directly related to the size of the hypothesis space, the used bias must have a comparable strength in both approaches. While symbolic approaches use this extra evidential knowledge to restrict the language to represent their hypotheses, the case-based algorithms need it to get appropriate measures of similarity.

Acknowledgement

We would like to thank M.M. Richter, K.P. Jantke, S. Lange, K.-D. Althoff, and H.-D. Burkhard for many helpful discussions. The study of case selection strategies in Section 4 was done together with Steffen Lange.

References

Aamodt, A., & Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, *7*(1), 39–59.

Aha, D. W. (1991). Case-Based Learning Algorithms. In Bareiss, R. (Ed.), *Proceedings: Case-Based Reasoning Workshop*, pp. 147 – 158. Morgan Kaufmann Publishers.

Angluin, D. (1980). Inductive inference of formal languages from positive data. *Information and Control*, *45*, 117–135.

Angluin, D., & Smith, C. H. (1983). Inductive Inference: Theory and Methods. *Computing Surveys*, *15*(3), 237–269.

Dasarathy, B. (1990). *Nearest Neighbor Norms: NN Pattern Classification Techniques*. IEEE Computer Society Press.

Globig, C., & Lange, S. (1994). On case-based representability and learnability of languages. In Arikawa, S., & Jantke, K. (Eds.), *Algorithmic Learning Theory*, Vol. 872 of *LNAI*, pp. 106–121. Springer-Verlag.

Globig, C. (1993). Symbolisches und Fallbasiertes Lernen. Masters Thesis, University of Kaiserslautern.

Globig, C., & Wess, S. (1994). Symbolic Learning and Nearest-Neighbor Classification. In Bock, P., Lenski, W., & Richter, M. M. (Eds.), *Information Systems and Data Analysis*, Studies in Classification, Data Analysis, and Knowledge Organization, pp. 17–27. Springer Verlag.

Gold, E. M. (1967). Language identification in the limit. *Information and Control*, *10*, 447–474.

Holte, R. S. (1990). Commentary on: PROTOS an exemplar-based learning apprentice. In Kodtratoff, Y., & Michalski, R. (Eds.), *Machine Learning: An Artificial Intelligence Approach*, Vol. III, pp. 128–139. Morgan Kaufmann.

Jantke, K. P. (1992). Case-Based Learning in Inductive Inference. In *Proceedings of the 5th ACM Workshop on Computational Learning Theory (COLT'92)*, pp. 218–223. ACM-Press.

Jantke, K. P., & Lange, S. (1989). Algorithmisches lernen. In Grabowski, J., Jantke, K. P., & Thiele, H. (Eds.), *Grundlagen der Künstlichen Intelligenz*, pp. 246–277. Akademie–Verlag, Berlin.

Jantke, K., & Lange, S. (1993). Case-based representation and learning of pattern languages. In *Proceedings of the 4th International Workshop on Algorithmic learning Theory (ALT'93)*, Vol. 744 of *LNAI*, pp. 87–100. Springer-Verlag.

Kolodner, J. L. (1993). *Case-Based Reasoning*. Morgan Kaufmann.

Michalski, R., Carbonell, J. G., & Mitchell, T. (Eds.). (1983). *Machine Learning: An Artificial Intelligence Approach*, Vol. 1. Tioga, Palo Alto, California.

Mitchell, T. (1982). Generalization as search. *Artificial Intelligence*, *18*(2), 203–226.

Rendell, L. (1986). A General Framework for Induction and a Study of Selective Induction. *Machine Learning*, *1*, 177–226.

Richter, M. M. (1992). Classification and Learning of Similarity Measures. In *Proc. der 16. Jahrestagung der Gesellschaft für Klassifikation e.V.* Springer Verlag.

Wess, S., & Globig, C. (1994). Case-based and symbolic classification – a case study. In Wess, S., Althoff, K.-D., & Richter, M. (Eds.), *Topics in Case-Based Reasoning*, Vol. 837 of *Lecture Notes in Artificial Intelligence*, pp. 65–76. Springer-Verlag.

Wess, S. (1993). PATDEX - Inkrementelle und wissensbasierte Verbesserung von Ähnlichkeitsurteilen in der fallbasierten Diagnostik. In *Tagungsband 2. deutsche Expertensystemtagung XPS-93*, pp. 42–55 Hamburg. Springer Verlag.