Case-Based and Symbolic Classification Algorithms*
- A Case Study using Version Space -

Stefan Wess and Christoph Globig

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

Abstract. Contrary to symbolic learning approaches, that represent a learned concept *explicitly*, case-based approaches describe concepts *implicitly* by a pair (CB, sim), i.e. by a measure of similarity sim and a set CB of cases. This poses the question if there are any differences concerning the learning power of the two approaches. In this article we will study the relationship between the case base, the measure of similarity, 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 hypothesis 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 article we want to compare the learning power of two important learning paradigms – the *symbolic* and the *case-based* approach [1, 4]. As a first step in this direction, Jantke [9] has already analyzed the common points of inductive inference and case-based learning. Under the term *symbolic learning*[2] we subsume approaches, e.g. [12], 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. The learning task we want to study is the classification of objects. The aim of a classification task is to map the objects x of a universe U to 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 certain concept C, i.e. the universe U is separated in two $C, \neg C$ disjoint subsets.

* The presented work was partly supported by the *Deutsche Forschungsgemeinschaft*, SFB 314: "Artificial Intelligence and Knowledge-Based Systems" and the project IND-CBL.

[2] 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.

For this special scenario we will show that case-based approaches have the same learning power as symbolic approaches. We will introduce two different approaches to case-based classification and show that the results achieved in this work are independent from the used case-based approach. After that we will present a simple symbolic learning algorithm (the Version Space [13]) and transform this algorithm into a case-based variant to show that the symbolic and the case-based algorithm are equivalent in the learning power. Based on this example we will show that for case-based approaches there exists a strong tradeoff between the set of learnable concepts and the minimal number of cases in the case base. We will conclude that for our scenario the used bias must have a comparable strength in both approaches.

The fundamental problem the two approaches have to solve during the learning phase is the same. At every moment the learner 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.

2 Basic Algorithm for Case-Based Classification

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 $(x, class(x))$ where x is a description of the case and $class(x)$ is the classification. Given a new case $(y, ?)$ with unknown classification, the system searches in the case base CB for the nearest neighbor $(x, class(x))$ (or the most similar case) according to a given measure of similarity sim. Then it states the classification $class(x)$ of the nearest neighbor as the classification of the new case $(y, ?)$, i.e $(y, class(x))$. The basic algorithm [1, 3] for a case-based approach is presented in Fig. 1.

Basic Algorithm for Case-Based Classification

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

Fig. 1. Basic Algorithm for a Case-Based Classifier

From the viewpoint of machine learning, case-based learning may be seen as a *concept formation task*. 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* [8] by a pair (CB, sim). The relationship between the case base and the measure used for classification may be characterized by the equation:

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

This equation indicates in analogy to arithmetic that it is possible to represent a given concept C in multiple ways, i.e. there exist many pairs $C = (CB_1, sim_1)$, $(CB_2, sim_2), \ldots, (CB_k, sim_k)$ for the same concept C. Furthermore, the equation gives a hint how a case-based learner can improve its classification ability. There are three possibilities to improve a case-based system. The system can

- store new cases in the case base CB,
- change the measure of similarity sim,
- or change CB and sim.

During the learning phase a case-based system (cf. Fig. 1) gets a sequence of cases $X_1, X_2, \ldots, X_k$ with $X_i = (x_i, class(x_i))$ and builds up a sequence of pairs $(CB_1, sim_1), (CB_2, sim_2), \ldots, (CB_k, sim_k)$ with $CB_i \subseteq \{X_1, X_2, \ldots, X_i\}$. The aim is to get in the limit a pair (CB_n, sim_n) that needs no further change, i.e. $\exists n\ \forall m \geq n\ (CB_n, sim_n) = (CB_m, sim_m)$, because it is a correct classifier for the target concept C.

Case-based systems apply techniques of nearest-neighbor classification [6] in symbolic domains. The basic idea is to use the knowledge of the known cases directly to solve new problems. By directly we mean, that the case-based system does not try to extract explicit knowledge during the learning phase and apply this abstract knowledge during the application phase.

2.1 Methods for Retrieval

The most important point in a case-based approach is the method to retrieve a similar case. Basically, there are two main approaches:

Representational Approach: In the representational approach [11] the method for retrieving the similar case is coded into the structure of the case base itself. The cases are connected by index structures. In order to determine the most similar case the index structure is traversed. Cases being neighbors according to the used index structure are stated as similar.

Computational Approach: In the computational approach, e.g. memory-based reasoning [17] or instance-based (resp. case-based) learning [1, 3], the known cases are stored as an unstructured set (Fig. 1). The most similar case is determined by the evaluation of a similarity measure $sim\colon U \times U \rightarrow [0, 1]$.

The dual notion is that of a *distance measure* $d\colon U \times U \to \mathcal{R}^+$. In the sequel we will use the term *measure* if we do not want to distinguish between similarity and distance measures. Both types of measures have the same power [16] and we will use them with respect to the context of the examples. Usually, the specific values of sim or d are of less interest than the relations between them [16, 15]. The following relation *y is more similar to x than v to u* is important for the computational approach:

$$R_{sim}(x,y,u,v) : \Longleftrightarrow\ sim(x,y) \geq sim(u,v)$$
$$R_d(x,y,u,v) : \Longleftrightarrow\ d(x,y) \leq d(u,v)$$

We say that d and sim are *compatible*, iff

$$R_d(x,y,u,v) \iff R_{sim}(x,y,u,v)$$

For the retrieval of similar cases we define (R_d and R_{sim} are compatible)

$$S(x,y,z) \iff R_d(x,y,x,z) \iff R_{sim}(x,y,x,z)$$

Some $y \in U$ is called *most similar* to x with respect to U iff

$$\forall z \in U\ S(x,y,z)$$

This relation $S(x,y,z)$ provides a partial-ordering $\succeq$ on the tuples of cases of U.

$$S(x,y,z) \iff R(x,y,x,z) \iff (x,y) \succeq (x,z) \iff y \succeq_x z$$

We will call such an ordering a *preference relation* [15]. Each case x of the universe U induces a specific preference relation $\succeq_x$. Thereby, $y \succeq_x z$ means that *the case y is more appropriate to solve the classification task x than the case z.*

2.2 Comparison

The computational as well as the representational approach have to construct such a partial-ordering $(CB, \succeq)$ of all cases in the case base. Instead of using a distance or a similarity measure itself, in the representational approach the relation $S(x,y,z)$ is represented by the used index structure.

We have now to analyze the question, whether the method to compute the preference relation $\succeq$ has any influence on the result of the retrieval process. To answer the question suppose $\succeq_y \ \ (y \in U)$ is any realization of the relation S. Suppose further that the system will retrieve exactly one case from the case base to solve the new problem. Then we can define a measure of similarity $sim_{pre} : U \times U \to \{0,1\}$ by

$$sim_{pre}(x,y) = \begin{cases} 1 \text{ if } x \text{ is maximal in } (CB, \succeq_y) \\ 0 \text{ otherwise} \end{cases}$$

If we now compare $\succeq_y$ and $sim_{pre}(\cdot, y)$ on the same case base, it is obvious that both methods will retrieve the same cases. To find the most similar case to y with

$\succeq_y$, we search for the maximal elements of $(CB, \succeq_y)$. If we use sim_{pre}, we search for cases that are maximal similar to the given case. Because of $sim_{pre}(x, y)$ is maximal if x is a maximal element of $(CB, \succeq_y)$, it follows that every realization of a preference relation $\succeq$ can be described by a numerical measure of similarity.

With this method we are now able to transform every given preference relation into a compatible measure of similarity. So we can conclude that the result of the retrieval process is not affected if the paradigm of computing the preference relation $\succeq$ is changed. Thus, the results of this work can be applied to case-based systems that follow either the representational or the computational approach to case retrieval. For reasons of simplicity, we will restrict ourselves in the sequel to case-based systems that realize the preference relation $\succeq$ by a measure of similarity sim or a measure of distance d.

3 A Case-Based Variant of a Symbolic Learner

To demonstrate the fundamental equivalence of the learning power of symbolic and case-based learners, we transform a well-known symbolic learner – the Version Space (VS) from [13] – in an equivalent 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 a lot of properties, which hold for many other learning algorithms, where it would be difficult to prove them.

3.1 The 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[3]. We can describe these concepts C as vectors $(C_1, \ldots, C_n)$, with $C_i = *$ or $C_i = a_{ij} \in W_i$. A case $((a_1, \ldots, a_n), class(a))$ 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. $class(x) = +$, of the set fulfill the concept and none of the negative, i.e. $class(x) = -$, does. The symbolic version space [13] 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 their properties. Figure 2 shows the algorithm (cf. [13]). 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

[3] i.e. these concepts represent the conjunctions of atomic formulas $x_i = a_i$, e.g. $shape = circle \wedge size = big$.

Version Space Algorithm

1. Initialize $G = \{(*, ..., *)\}$ and $S = \{a^1\}$.
2. If the actual case $(a, class(a))$ is positive 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 $(a, class(a))$ is negative then
   remove all concepts from S which are not fullfilled by a.
   For all concepts $g \in G$, that subsume a, search for the most general specializations, that do not subsume a but all known positive cases.
   Replace g by the found concepts.
4. If G or S is empty or there is a concept 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.

Fig. 2. Algorithm for the symbolic version space

negative. This observation leads to a partial decision function $VS : U \rightarrow \{0, 1\}$ that can be used to classify new cases:

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

As long as $S \neq G$ VS will not classify all cases of the universe. If an case is covered by S but not by G it is not clear whether it belongs 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).

Example To illustrate this version space algorithm we present a very simple example. The universe U is $U =$ shape $\times$ size $=$ {square, circle} $\times$ {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:

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

Let the second case be negative $((square, small), -)$. This forces us 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

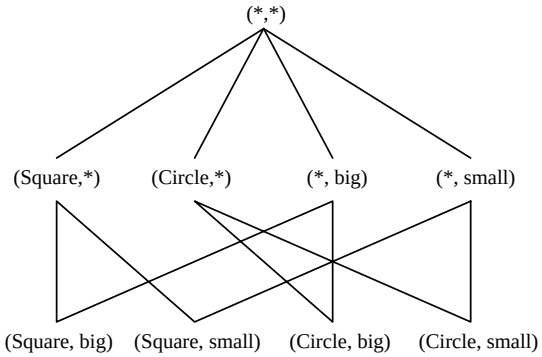


Fig. 3. Set of the learnable concepts over U

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 A Case-Based Variant of the 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 $C = (a_1, \ldots, a_n)$. For every attribute i, we define a function f_i that maps $x \in W_i$ to $\{0, 1\}$ with the following definition:

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

The functions f_i will be combined to $f : U \rightarrow \{0,1\}^n$ $f((a_1, \ldots, a_n)) = (f_1(a_1), \ldots, f_n(a_n))$. The distance between two cases a and b is then defined using the city-block metric as follows

$$d_f(a, b) := |f_1(a_1) - f_1(b_1)| + \ldots + |f_n(a_n) - f_n(b_n)|$$

It is obvious that every change of the functions $f_1, f_2, \ldots, f_n$ causes a change of the underlying measure d_f. The intended function f_i is learnable by the algorithm in Fig. 4. The algorithm expects the first case to be positive.
If the concept is learned, the function f and the case base CB are used for classification. Given a new case $(c, ?)$, the set

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

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 $((a_1, \ldots, a_n), +)$. Let $f_i(a_i) = 1$ and $CB = \{(a, +)\}$
3. Get a new case $((b_1, \ldots, b_n), class(b))$.
4. If $class(b)$ is negative, store b in the case base CB, i.e. $CB := CB \cup \{(b, -)\}$
5. If $class(b)$ 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 $(a, class(a)), (b, class(b)) \in CB$ with $d_f(a, b) = 0$ and $class(a) \neq class(b)$ 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_f) consists of the case base CB and the measure d_f
8. Go to step 3.

Fig. 4. Algorithm to learn f for VS-CBR

is computed. The classification $class(x)$ of the most similar case $(x, class(x))$ is then used for the classification of the new case $(c, ?)$. If F contains more than one case and these cases have different classifications then $class(c)$ is determined by a fixed strategy to solve this conflict. Different strategies are possible and each strategy will induce a own decision function 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:

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

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

Example Before we analyze 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). Because our universe has only two dimensions, we need two functions $f_1 : shape \rightarrow \{0, 1\}$ and $f_2 : size \rightarrow \{0, 1\}$. The first positive case $((circle, big), +)$ is used to initialize the functions f_1 and f_2.

$$f_1(x) = \begin{cases} 1 \text{ if } x = circle \\ 0 \text{ otherwise} \end{cases} \qquad f_2(y) = \begin{cases} 1 \text{ if } y = big \\ 0 \text{ otherwise} \end{cases}$$

$$CB = \{((circle, big), +)\}$$

The next case of our sequence is $((square, small), -)$. We have to store this new case in the case base.

$$f_1(x) = \begin{cases} 1 \text{ if } x = circle \\ 0 \text{ otherwise} \end{cases} \qquad f_2(y) = \begin{cases} 1 \text{ if } y = big \\ 0 \text{ otherwise} \end{cases}$$

$$CB = \{((circle, big), +), ((square, small), -)\}$$

Now (CB, d_f) will only classify $((circle, big), +)$ as positive, because every other case has a distance ≥ 1 from $((circle, big), +)$ and ≤ 1 from $((square, small), -)$. As third case assume $((square, big), +)$. Because $f_1(square) = 0$ we define f_1 to be zero for all values. We do not need to store this new case in the case base.

$$f_1(x) = 0 \qquad f_2(y) = \begin{cases} 1 \text{ if } y = big \\ 0 \text{ otherwise} \end{cases}$$

$$CB = \{((circle, big), +), ((square, small), -)\}$$

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 f, i.e. to adapt the distance measure d_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 d.

The information that is used by VS to change S and G is used by VS-CBR to change the case base or the measure of similarity. 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 it knows that the proposed classification must 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.

4 Relationships between CB, sim, and C

We have shown that it is possible to reformulate the Version Space algorithm in a case-based manner so that the case-based variant behaves as the symbolic algorithm. It is important to understand the implications of a measure of similarity to the set of representable concepts. In the sequel, 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
   $\forall a, b, x \in U\, [d(a,a) = 0 \wedge d(a,b) = 0 \Rightarrow d(x,a) = d(x,b)]$

Condition 3 has two important consequences: First, the relation $\sim \subseteq U \times U$ defined by $x \sim y \Leftrightarrow d(x,y) = 0$ 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 $d(x,y) = 0 \Rightarrow C(x) \equiv C(y)$, 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.

4.1 Dimensions of Comparison

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 be 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 *sim*. 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 *sim* will result in a smaller (minimal) size of the case base CB within the classifier (CB, sim).

Definition 1. The similarity measure sim_1 of a case-based system (CB_1, sim_1) is called *better informed* than a measure sim_2 of a system (CB_2, sim_2) iff both systems are classifiers for the same concept C, $|CB_1| < |CB_2|$ holds, and there is no $CB_i' \subset CB_i$ so that (CB_i', sim_i) is a classifier for the concept C.

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

Definition 2. A similarity measure sim_1 is called *more universal* than a similarity measure sim_2 iff the set of concepts that are learnable by sim_2 is a proper subset of the set of concepts that are learnable by sim_1.

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. We can distinguish two extreme situations:

All knowledge is coded into the case base: The similarity is maximal if and only if the compared cases are identical, i.e. $sim(x, y) := x = y$. The measure 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 $(U, =)$ is universal but not minimal.

All knowledge is coded into the measure: The similarity is maximal if and only if the classification of the compared cases is identical, i.e. the measure of similarity sim knows the definition of the concept C to learn. Nearly the whole knowledge about the concept is then coded into the measure. The case base contains almost one positive and one negative case and is used only to choose between some trivial variations. The measure $sim(x, y) := C(x) \equiv C(y)$ 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 $(\{c^+, c^-\}, C(x) \Leftrightarrow C(y))$ is minimal but not universal.

used measure	minimal size of CB	number of learnable concepts
$sim(x, y) := (x = y)$	$65536 = 16^4$	2^{65536}
VS-CBR$_1$	16	$65536 = 2^{16}$
VS-CBR$_2$	3 (4)	$16 = 2^4$
$sim(x, y) := (C(x) \equiv C(y))$	2	2^2

Fig. 5. Measures together with the minimal case base and the number of learnable concepts

We illustrate the contrasting nature of these two aims in Fig. 5. This figure shows different measures sim in relation to the minimal size of the case base CB to learn a certain concept C and 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 size of the universe U is 65536. The concepts C_i the measures try to learn, fix two attributes out of four. The other measures in the table 5 are between these two extremes. VS-CBR$_1$ and VS-CBR$_2$ are neither maximally universal nor able to represent the concept with a minimal case base. VS-CBR$_1$ 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. VS-CBR$_2$ 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.

In a case-based learner, two processes – reducing the size of the learnable concepts (hypothesis space) and increasing the size of the case base – should be performed. The last measure in Fig. 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.

4.2 The Choice of the Distance Measure

We have argued that the distance measure d is able to learn a concept C if and only if $d(x,y) = 0 \Rightarrow C(x) \equiv C(y)$. Whether a concept C is learnable by a given distance measure d then depends on the definition of the identity, i.e the distance $d(x,y) = 0$. If the concept C is learnable by d, all other distances may be mapped to any value greater than zero.

Does it make any sense to use a distance measure d that maps distances between cases to any other set than $\{0,1\}$? The only reason to use a more complex distance measure 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.

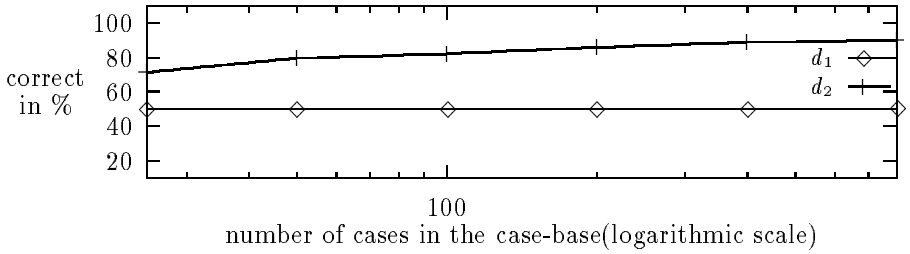
Example To illustrate this, we compare two measures that can learn the same concepts. Let the universe consist of cases with four attributes. The values for each attribute are $\{0,\ldots,15\}$. So the universe is $U = \{0,\ldots,15\}^4$. The concept to learn is $C(x),\ x := (x_1,\ldots,x_4)$.

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

and the distance measures are:

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

Obviously both measures are able to represent the same concepts in the universe U, because $d_i(a,b) = 0 \Leftrightarrow a = b$. Figure 4.2 shows the classification rates for the given concept $C(x)$ 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 of 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 [7]. 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:

$$C_2(x) = 1 \Leftrightarrow ((x_2 \mod 2 = 0) \wedge (x_4 \mod 2 = 1)) \vee ((x_4 \mod 2 = 0) \wedge (x_2 \mod 2 = 1))$$

The rate of correct classifications for d_2 and concept C_2 will be nearly the same as the rate of d_1 in Fig. 4.2.

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. The measure defines a *preference relation* $y \succeq_x z$ between two cases (cf. section 2.2) and is therefore independent from the existence of a concept. By transforming a given preference relation $\succeq$ into a measure $sim(x, y)$ we have shown (cf. section 2.2) that the results of this work can be applied to case-based systems that follow the *representational* or the *computational approach* to case retrieval.

A main difference between case-based and symbolic classification algorithms is the representation of the learned concept (cf. section 2). A case-based classifier (CB, sim) consists of a case base CB and a measure of similarity sim. It is possible to represent the same concept C in multiple ways, i.e. by different tuples (CB_i, sim_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 not a 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 4.1) 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 [9] 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 4.2), a case-based approach is a good choice. In this scenario we are able to code the neighborhood principle into the measure used. 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 4) the relationship between the measure of similarity, the case base, and the target concept in the described scenario of classification tasks. 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 have to code this assumptions into the measure of similarity. These restrictions of the hypothesis space are called *bias*. [14] 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 4.1) 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 [14]. 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.

We have stressed that the measure (respectively the way to modify the measure) is the *bias of case-based reasoning*. Because case-based systems are based on a bias that cannot be deduced from the cases, we reject the thesis [5] that case-based classification is more appropriate in situations with a low amount of previous knowledge. We conclude that for classification tasks there is no fundamental advantage in the learning power of case-based systems as maintained by Cost & Salzberg [5]. As we have seen (cf. section 4.2) 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. The thesis that intelligibleness is a property of the case-based approach is also stated by Cost & Salzberg [5]. Since the number of cases an algorithm

need 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.-D. Althoff, H.-D. Burkhard, and K.P. Jantke for many helpful discussions and the reviewers for their comments.

References

1. David W. Aha. Case-Based Learning Algorithms. In Ray Bareiss, editor, *Proceedings: Case-Based Reasoning Workshop*. Morgan Kaufmann Publishers, 1991.
2. David W. Aha and Dennis Kibler. Noise-Tolerant Instance-Based Learning Algorithms. In *Proceedings of the 11th International Conference on Artificial Intelligence IJCAI-89*, pages 794–799, 1989. Detroit, Michigan, USA.
3. David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-Based Learning Algorithms. *Machine Learning*, 6:37–66, 1991.
4. R. Bareiss, B. W. Porter, and C. C. Wier. PROTOS: An Exemplar-Based Learning Apprentice. In Kodratoff and Michalski [10], pages 12–23.
5. S. Cost and Steven Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Maschine Learning*, 10(1):56–78, 1993.
6. Belur Dasarathy. *Nearest Neighbor Norms: NN Pattern Classification Techniques*. IEEE Computer Society Press, 1990.
7. Christoph Globig and Stefan Wess. Symbolic Learning and Nearest-Neighbor Classification. In P. Bock, W. Lenski, and M. M. Richter, editors, *Proc. der 17. Jahrestagung der Gesellschaft für Klassifikation*. Springer Verlag, 1993.
8. Robert S. Holte. Commentary on: PROTOS an exemplar-based learning apprentice. In Kodtratoff and Michalski [10], pages 128–139.
9. Klaus P. Jantke. Case-Based Learning in Inductive Inference. In *Proc. COLT-92*, 1992.
10. Yves Kodratoff and Ryszard Michalski, editors. *Maschine Learning: An Artificial Inteligence Approach*, volume III. Morgan Kaufmann, 1990.
11. Janet L. Kolodner. *Retrieval and Organizational Strategies in Conceptual Memory*. Lawrence Erlbaum, Hillsdale, New Jersey, 1984.
12. R. Michalski, J. G. Carbonell, and T. Mitchell, editors. *Maschine Learning: An Artificial Intelligence Approach*, volume 1. Tioga, Palo Alto, California, 1983.
13. T.M. Mitchell. Generalization as search. *Artificial Intelligence*, 18(2):203–226, 19982.
14. L. Rendell. A general framework for induction and a study of selective induction. *Machine Learning*, (1):177–226, 1986.
15. Michael M. Richter. Classification and Learning of Similarity Measures. In *Proc. der 16. Jahrestagung der Gesellschaft für Klassifikation e.V.* Springer Verlag, 1992.
16. Michael M. Richter and Stefan Wess. Similarity, Uncertainty and Case-Based Reasoning in PATDEX. In Robert S. Boyer, editor, *Automated Reasoning*, Essays in Honor of Woody Bledsoe, pages 249–265. Kluwer Academic Publishing, 1991.
17. Craig Stanfill and David Waltz. Toward Memory-Based Reasoning. *Com. of the ACM*, 29(12):1213–1229, 1986.

This article was processed using the LaTeX macro package with LLNCS style