

Learning a Local Similarity Metric for Case-Based Reasoning

Francesco Ricci and Paolo Avesani

Istituto per la Ricerca Scientifica e Tecnologica
38050 Povo (TN)
Italy
email: {ricci,avesani}@irst.itc.it

Abstract. This paper presents a new class of local similarity metrics, called AASM, that are not symmetric and that can be adopted as the basic retrieval method in a CBR system. An anytime learning procedure is also introduced that, starting from an initial set of stored cases, improves the retrieval accuracy by modifying the local definition of the metric. The learning procedure is a reinforcement learning algorithm and can be run as a black box since no particular setting is required. With the aid of classical test sets it is shown that AASM can improve in many cases the accuracy of both nearest neighbour methods and Salzberg's NGE. Moreover, AASM can achieve significant data compression (10%) while maintaining the same accuracy as NN.

1 Introduction

Classification methods based on nearest neighbor (NN) have many advantages compared with other classification techniques. First of all, NN supports incremental learning from new cases without degradation in performance on previous training data. Moreover, NN methods work with an order of magnitude fewer parameters than back-propagation or radial basis function methods and are quite straightforward to implement.

Nevertheless classical NN methods have some negative points that limit their applicability. Classical NN methods exhibit poor generalization performance, suffer from the existence of noisy features, require big training sets and they suffer from the so called "curse of dimensionality". As the training set grows the run time performance of the system increases (see [9] for a large collection of papers on NN and his generalizations). These limitations have been addressed positively by a number of papers, which will be partially reviewed here, and are also the topic of our work.

NN methods are at the base of many implementations of case-based reasoning systems [25, 3, 24]. A training set in NN terminology is also called a case base, and an input to be classified is also called a probe case to be matched with the case base by the retrieval function. Both terminologies always will be adopted to stress the similarities between the two research areas. In a case-based reasoning framework our main goals may be stated as follows:

- Reduce the memory space required by the case base. The objective is to filter out cases in such a way that the accuracy of the system is not decreased (data compression).
- Speed-up on-line queries to the case base. This goal is related in some way to the previous one. NN algorithms usually have a linear time complexity with respect to the dimensionality of the case base. Therefore a reduction of the case base yields a proportional reduction in response time¹.

Researchers have recognized that many maladies of NN arise from the choice of the similarity metric, which is normally the classical Euclidean metric. First global modifications of the Euclidean metric have been considered and then a few researchers proposed local definition of the similarity metric [4, 5, 23]. A *global metric* is here defined as a simple generalization of the Euclidean metric, in this case a vector of weights is used to balance the contributions of different distances on the axes ($d(x, y) = (\sum_{i=1}^N w_i |x_i - y_i|^2)^{1/2}$). A *local metric* is a metric that depends on the point in the input space from which the distance is taken ($d(x, y) = (\sum_{i=1}^N w_i(x) |x_i - y_i|^2)^{1/2}$). Local metrics are context-sensitive [6], that is, similarity between cases depends also on the absolute value of feature values. Using a local metric one can express conditional expressions like “if feature A is greater than 70% then use only feature B and C to compute similarity”. For instance, a local metric was used in a complex planning problem [7, 20] because an analysis of the domain knowledge raised the need of a context-sensitive similarity measure.

One of the basic tenets of our approach is that a local metric, which has been adapted to the problem domain, can lead to a significant reduction of the number of stored cases (data compression) and therefore to a substantial speed-up in run time performance. Moreover, adapting a local metric to an input space can also reduce the degradation of accuracy that is often associated with the introduction of irrelevant features.

This paper introduces a novel approach to compute nearest neighbor based on a local metric called AASM (asymmetric anisotropic similarity metric). This approach makes two basic assumptions. The first one (anisotropic) states that the metric is defined locally: the space around a case in memory is measured using the metric attached to that case. The second one (asymmetric) states that the distance between two points in a continuous feature space F_i is not symmetric, i.e., $d_i(x_i, y_i) \neq d_i(y_i, x_i)$. In fact two different weights for the “left” and the “right” directions are used per feature. In this way, as it is shown in a next section, one can more freely choose a representative in a set of cases that have to be commonly classified.

A reinforcement learning procedure [17, 14] also is provided for adapting the local weights to the input space. Our model basically implements an anytime algorithm that given an input case c decreases the distance between c and the nearest neighbor nn if nn correctly classifies c (reinforcement), and increases that distance if nn does not correctly classify c (punishment). A set of experiments

¹ Speed up can also be achieved compiling the case base in a k-d tree [25].

shows how a drastic reduction of the case base can be achieved, still maintaining the same accuracy of other methods.

Among the advantages of this new approach is the fact that AASM can be run as a black box without setting problem-specific parameters. Another advantage is that AASM can be assigned a fixed amount of memory and a fixed time to reply to a query. The system performs better and better as the amount of memory and the time to reply are increased. A drawback of our model is that for each case we need to store two additional vectors of the same dimension of the input space. We claim that this additional memory space is paid for by: a reduction of the cases needed to obtain the same accuracy of NN; an improvement in run time performance.

2 Previous Research

There are a number of generalizations of the basic nearest neighbor algorithm that address one of the points raised above: local metric, metric adaptation and case base reduction (data compression). A more extensive discussion of similar approaches and in particular of Learning Vector Quantization [13] can be found in [19].

Lowe [15] adopts a global metric and an optimization technique based on conjugate gradient method to optimize the similarity metric and the width of a Gaussian Kernel that is used to balance the contributions of k nearest neighbors. This method can cope with noisy features, but like others based on a global metric, it assumes that the noisy features are always the same in all the regions of the input space.

Cost and Salzberg [22, 8] use a global metric with an additional weight for each stored case, that measures how frequently the case was used to make a correct classification (prediction). Some cases are "generalized" by the learning process, i.e., they are replaced by the minimal hyperrectangle that contains that point. A hyperrectangle in $[0, 1]^n$ is defined as: $H_{xy} = \{z \in [0, 1]^N \mid z_i \in [\min(x_i, y_i), \max(x_i, y_i)] \forall i = 1, \dots, N\}$. The distance between a point and a hyperrectangle is defined as the distance between a point and a set in Euclidean geometry, so for example if a point is inside a hyperrectangle the distance between them is zero. NGE uses a global definition of the metric weights but the metric is locally influenced by the dimension of hyperrectangles. Experimental results reported here, which were run on the same data sets used in [22], compare the accuracy achieved by EACH's learning procedure (Exemplar-Aided Constructor of hyperrectangles) with ours. A somewhat similar approach to generalization, which is exploited for rule induction, is also presented in [11].

Aha and Goldstone [4, 5] claim that an attribute's importance in human classification depends on its context and they provide a computational model that is able to generalize with results similar to those shown by experiments conducted with humans. One of the computational models proposed by the authors (GCM-ISW) uses an interpolation of local and global metrics. They show that GCM-ISW provides a better fit to the subject data than other methods with no

local weights. Aha also introduced in [1, 2] an instance-based learning algorithm (IB4) that adopts a global metric and a non-parametric reinforcement learning strategy to incrementally change the weights. IB4's classification of an instance is decided by a vote among the k most similar instances and matches of instances in less frequent concepts yield larger adjustments of attribute weights.

3 Anisotropic and Asymmetric Metrics

Let C be a case space, $C = F_1 \times \dots \times F_N$, where each F_i is the unit interval $[0, 1]$ or a set of symbols. Let us define a distance $d_i : F_i \times F_i \longrightarrow \mathbb{R}_{\geq 0}$ on each F_i with the following equation:

$$d_i(x_i, y_i) = \begin{cases} |x_i - y_i| & \text{if } F_i = [0, 1] \\ 0 & \text{if } F_i \text{ is a set of symbols and } x_i \neq y_i \\ 1 & \text{if } F_i \text{ is a set of symbols and } x_i = y_i \\ 0.5 & \text{if } x_i \text{ or } y_i \text{ is unknown} \end{cases}$$

These metrics on F_i spaces can be extended to a metric on C as usual: $d(x, y) = (\sum_{i=1}^N d_i(x_i, y_i)^n)^{1/n}$, where n is an integer greater than or equal to 1. This is the usual L^n metric on the product space C .

We shall now modify the Euclidean metric relaxing the symmetric condition ($d(x, y) = d(y, x)$) and enabling a different definition of the metric in different points of the case space. Let us first relax the symmetry of the metric d , we define a new metric $\delta_i : F_i \times F_i \longrightarrow \mathbb{R}_{\geq 0}$

$$\delta_i(x, y) = \begin{cases} p_i(x - y) & \text{if } x \geq y \\ q_i(y - x) & \text{if } x < y \end{cases}$$

where $p_i, q_i \in [0, 1]$ and $F_i = [0, 1]$. These two parameters are called the *left* and *right* weights of the i -th feature. A weighted metric can be defined also on a generic set of symbols but in this case an order relation could not always be defined, so we simply pose:

$$\delta_i(x, y) = w_i d_i(x, y)$$

where $w_i \in [0, 1]$. Let us suppose \overline{C} be a subset of C . \overline{C} represents the set of cases stored in memory. $\overline{C} = \{x_1, \dots, x_M\}$ and for each $x_i = (x_{i1}, \dots, x_{iN}) \in \overline{C}$ let $p_i = (p_{i1}, \dots, p_{iN})$, and $q_i = (q_{i1}, \dots, q_{iN})$ be two vectors in $[0, 1]^N$. Let us further suppose that $p_i = q_i$ if F_i is a set of symbols. The two matrices (p_{ij}) and (q_{ij}) are called a *System of Weights* for the case base \overline{C} . In other words a system of weights is a point in $[0, 1]^{2N|\overline{C}|} = W$.

We can now define an anisotropic and asymmetric metric on $\overline{C} \times C$, such that:

$$\delta : \overline{C} \times C \longrightarrow \mathbb{R}_{\geq 0}$$

$$\delta(x_i, y) = \left(\sum_{j=1}^N w_{ij} d_j(x_{ij}, y_j)^n \right)^{1/n}.$$

where

$$w_{ij} = \begin{cases} p_{ij} & \text{if } x_{ij} \geq y_j \text{ and } F_j = [0, 1] \\ q_{ij} & \text{if } x_{ij} < y_j \text{ and } F_j = [0, 1] \\ p_{ij} = q_{ij} & \text{if } F_j \text{ is a set of symbols} \end{cases}$$

So δ measures the distance between a point in the case base and a generic input. The following almost considers L^2 metrics, so we shall assume that $n = 2$ unless otherwise stated. The *retrieval function* (projection onto \overline{C}) on a case base C is defined as follows:

$$R_{\overline{C}}: C \longrightarrow \overline{C}$$

$$R_{\overline{C}}(y) = \overline{x} \text{ s.t. } \delta(\overline{x}, y) \text{ is minimum in } \overline{C}.$$

The retrieval function implements the basic nearest neighbor method: match a probe case with the most similar case in memory. Figure 1 shows a set of level

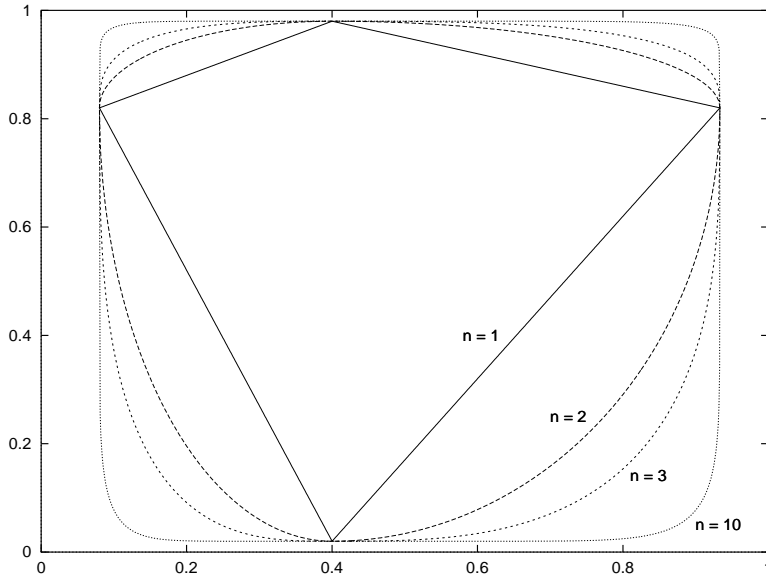


Fig. 1. Level curves for different values of p

curves for different values of n in a two dimensional space. Each curve is the set of points with distance 0.08 from point $(.4, .82)$ with different values for n . The weights chosen for the L^1 metric are $p_1 = .25$, $p_2 = 0.1$, $q_1 = 0.15$ and $q_2 = 0.5$. For $n = 2$ the weights are the square of the weights chosen for $n = 1$, for $n = 3$ they are the cube, and for $n = 10$ the weights are the tenth power of the weights chosen for $n = 1$. It is clear that for $n \rightarrow \infty$ the level curves tend to be rectangular².

² In fact L^∞ has the norm of the maximum: $d(x, y) = \max_i |x_i - y_i|$.

The retrieval function is generally used in a case-based system for approximation or classification purposes. Let D be a finite space³, and let $G : C \rightarrow D$ be a *goal function*. Let us suppose that G is known on \overline{C} , that is we know a correct solution for each case in \overline{C} . Here we want to approximate G on all the space C using its known definition on \overline{C} and the retrieval function R . Let \overline{G} be defined:

$$\overline{G} : C \rightarrow D$$

$$\overline{G}(y) = G(R_{\overline{C}}(y))$$

In other words \overline{G} maps a case y to the point (class/solution) that is mapped by the most similar case to y . Let us define the *accuracy* of a retrieval function $A = Prob[\overline{G}(y) = G(y)]$. If C is finite $A = |\{y \in C, \overline{G}(y) = G(y)\}|/|C|$.

Allowing different weights for left and right directions in a local metric results in a more free choice of the cases in \overline{C} . Let us illustrate this point with a very simple example. Let us suppose $C = [0, 1]$, $D = \{0, 1, 2, 3\}$, $0 = a_0 < a_1 < a_2 < a_3 < a_4 = 1$ are five points in $[0, 1]$ and define $G(x)$ as the greatest k such that $a_k \leq x$ if $x \in [0, 1[$ and $G(1) = 3$. Suppose now that the case base \overline{C} is composed of four points $\{x_1, \dots, x_4\}$ with $a_{k-1} < x_k < a_k$. Using a symmetric metric we can have accuracy 1 provided that the four weights $\{w_1, \dots, w_4\}$ (one for each point in $\{x_1, \dots, x_4\}$) satisfy the three equations: $w_1(x_1 - a_1)^2 = w_2(x_2 - a_1)^2$, $w_2(x_2 - a_2)^2 = w_3(x_3 - a_2)^2$, $w_3(x_3 - a_3)^2 = w_4(x_4 - a_3)^2$. But, if for some reason w_1 changes, all the other weights have to change accordingly. Therefore, local adaptation of the weights is impossible. Conversely, using an asymmetric local metric the three equations on the weights become: $q_1(x_1 - a_1)^2 = p_2(x_2 - a_1)^2$, $q_2(x_2 - a_2)^2 = p_3(x_3 - a_2)^2$, $q_3(x_3 - a_3)^2 = p_4(x_4 - a_3)^2$. It is clear that in this case the weights are not linked as in a chain, and for example a change of q_1 requires only adaptation of p_2 . Moreover in this case it is simpler to find out a “correct” set of weights because we have doubled the variables (weights) still maintaining the same number of constraints.

4 Learning Weights

We shall now present a procedure that, starting from a case base \overline{C} and a system of weights w , iteratively changes the weights in w . The goal is to improve the accuracy of retrieval computed with respect to a given goal function G . A *learning step* is a map

$$T : W \times C \rightarrow W$$

$$T : (p(n), q(n), y) \mapsto (T(p(n), y), T(q(n), y))$$

that maps a system of weights, defined at an instant n and a probe y in a new system of weights $(p(n+1), q(n+1))$ (see also [18] for more details on

³ The extension to the approximation problem in infinite spaces will be considered in another forthcoming paper.

learning steps and learning procedures based on reinforcement). In the following, if $w = (p, q)$ is a system of weights we shall also indicate with $w' = (p', q')$ a system of weights obtained from w applying the transformation T . A *learning procedure* is an algorithm that iteratively chooses a case and calls the learning step on it until an exit condition is satisfied.

The following illustrates a particular learning step we have adopted and the results of our experiments. All our experiments adopt a very simple linear reinforcement scheme [17].

Let $w = (p, q)$ be a system of weights, $y \in C \setminus \overline{C}$ a probe case and $(p', q') = T(p, q, y)$. We shall now define the learning step. There may be two cases:

1. If $\overline{G}(y) = G(y)$ then we have:

$$p'_{ij} = T_{ij}(p_{ij}, y_j) = \begin{cases} p_{ij} - \alpha p_{ij} |x_{ij} - y_j| & \text{if } x_{ij} \geq y_j \\ p_{ij} & \text{if } x_{ij} < y_j \end{cases}$$

$$q'_{ij} = T_{ij}(q_{ij}, y_j) = \begin{cases} q_{ij} & \text{if } x_{ij} \geq y_j \\ q_{ij} - \alpha q_{ij} |x_{ij} - y_j| & \text{if } x_{ij} < y_j \end{cases}$$

if $F_i = [0, 1]$.

$$w'_{ij} = T_{ij}(w_{ij}, y_j) = \begin{cases} w_{ij} - \alpha w_{ij} & \text{if } x_{ij} \neq y_j \\ w_{ij} & \text{if } x_{ij} = y_j \end{cases}$$

if F_i is a set of symbols.

2. If $\overline{G}(y) \neq G(y)$ then we have:

$$p'_{ij} = T_{ij}(p_{ij}, y_j) = \begin{cases} p_{ij} + \frac{\beta(1-p_{ij})}{1+|x_{ij}-y_j|} & \text{if } x_{ij} \geq y_j \\ p_{ij} & \text{if } x_{ij} < y_j \end{cases}$$

$$q'_{ij} = T_{ij}(q_{ij}, y_j) = \begin{cases} q_{ij} & \text{if } x_{ij} \geq y_j \\ q_{ij} + \frac{\beta(1-q_{ij})}{1+|x_{ij}-y_j|} & \text{if } x_{ij} < y_j \end{cases}$$

if $F_i = [0, 1]$.

$$w'_{ij} = T_{ij}(w_{ij}, y_j) = \begin{cases} w_{ij} + \frac{\beta(1-w_{ij})}{2} & \text{if } x_{ij} \neq y_j \\ w_{ij} + \beta(1-w_{ij}) & \text{if } x_{ij} = y_j \end{cases}$$

if F_i is a set of symbols.

$\alpha \in [0, 1]$ and $\beta \in [0, 1]$ are called the *reinforcement* and *punishment* rate respectively. We note that each learning step updates at most N parameters and maintains the weights in $[0, 1]$. We can now define the learning procedure as a simple loop that after having initialized the system of weights⁴, randomly chose a point in $(C \setminus \overline{C})$ (in fact we split $(C \setminus \overline{C})$ in two parts, one for training and another for test) and applies the T transformation on the system of weights using this point. The loop is terminated when an exit condition is satisfied. In all the experiments the process was stopped after a number of cycles proportional to the cardinality of \overline{C} was completed.

⁴ In all the experiments performed the weights were initially equal to .00001.

5 Experimental Results

We have conducted a set of experiments on four popular data sets[16]:

1. **Fisher’s Iris Data Set.** This data set consists of four measurements made by E.Anderson on 150 samples of three species of iris [12]. Each example consists of four real-valued variables plus a known assignment of the example to a specie. This database doesn’t have attributes with unknown values.
2. **Cleveland Data Set**⁵. The Cleveland data set contains cardiological diagnoses [10]. The experiments with this data set aim to assess the occurrence of heart diseases. Each patient’s record consists of 13 features (5 real-valued, 5 symbolic-valued, 2 boolean-valued). The goal is to distinguish the presence of heart diseases from the absence. The 303 instances are equally divided into the two classes.
3. **Breast Cancer Data Set**⁶. This database contains 286 cases of patients who have been operated for tumor removal. Each example contains nine variables (6 symbolic-valued, 3 boolean-valued) that were measured plus a binary prediction: either the patient suffered a recurrence of cancer (30%) or not (70%).
4. **Echocardiogram Data Set.** Each example is a record for a patient who has had a heart attack. Some are still alive and some are not. The problem addressed by past researchers was to predict whether or not the patient will survive at least one year. The thirteen predictive attributes are all real-valued with meaningful occurrences of unknown values.

Fisher’s Iris, Breast Cancer and Echocardiogram data sets were used, among others, by Salzberg to test NGE [22]. Wettschereck [26] compares a number of learning algorithms on different data sets, Fisher’s and Cleveland data sets are two of those.

Table 1. Comparison of the accuracy, average memory size and average time to test obtained on different data sets by different algorithms

Data Set	Algorithm								
	AASM			EACH			NN		
Breast Cancer	66.0	20	1.73	59.3	41.8	2.26	65.0	200	4.95
Iris	92.8	10	0.30	93.8	11.6	0.26	93.0	102	0.99
Echocardiogram	70.0	5	0.11	63.1	10.3	0.13	63.0	52	0.30
Cleveland	76.5	21	3.21	58.5	35.6	3.46	76.0	212	10.01

⁵ The data have been provided by Robert Detrano from the V.A. Medical Center, Long Beach and Cleveland Clinic Foundation.

⁶ The data have been provided by M. Zwitter and M. Soklic from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia.

For each data set 100 trials were executed subdividing the database into two parts: one for training, with 70% of the total database, and the other 30% for test. For each trial we have chosen a new random partition of the database and a different random set of seeds. The number of seeds both for NGE and AASM has always been taken as 10% of the training part and the seeds have been generated in such a way that each class has the same number of representatives in the set of seeds. The learning process in AASM was stopped after $10 * |TrainingSet|$ iterations of the learning step.

The results of these experiments⁷ are shown in Table 1 and in Figure 2. Table 1 shows the accuracy obtained with different algorithms, the number of cases that were introduced into the memory ($|\overline{C}|$) and the average time for testing the classifier. The memory size does not change during learning for AASM, whereas it is equal to the final number of hyperrectangles generated by EACH algorithm. Our implementation of EACH is based on the generalization given in [26] and the parameters of the algorithm are set as Salzberg does in [22] ($\Delta_f = 0.2$). In particular the area of hyperrectangles is used to break ties when an input case has the same distance from two hyperrectangles⁸. Another important generalization concerns unknown values. In fact the original algorithm does not deal with both unknown features and symbolic features. The symbolic features are treated as in [26], but we decided to measure differently feature distance in the presence of unknown values. Our approach is similar to that used by Aha in [2], namely to assign feature distance equal to 0.5 when a feature value is unknown. While running our experiments we learned that EACH has to be tailored to the data set, and the configuration given by Salzberg is not always the best one. For example a little improvement in performance can be obtained in the Iris data set without updating the weights.

AASM has greater accuracy than NN in all data sets but Iris and uses 10% of the memory used by this last method. The comparison with NN is more clearly shown in Figure 2. Here the ratio of the accuracy obtained by AASM and that of NN is shown, with increasing values of the average number of times a training sample is presented to AASM. For Iris and Echocardiogram data sets AASM reaches (after 5 average sample presentations) more than 95% of the accuracy of NN, that stores in the memory all the training component. In the Breast Cancer and Cleveland data set AASM reaches a better accuracy than NN, only after each sample in the trial is shown a couple of times on average.

AASM depends on two parameters, the reward and punishment factors α and β . A number of experiments were conducted with different pairs of values, but no significant differences were tested. All the data shown here were obtained with $\alpha = 0.2$ and $\beta = 0.1$. No particular procedure was used for reducing the reward and punishment parameters as learning proceeds, but we expect to

⁷ The experiments have been conducted using a public domain software written in LISP that is Copyright of Raymond J. Mooney (University of Texas at Austin). It includes automatic testing software for running learning curves that compare multiple systems and utilities for plotting and statistically evaluating the results.

⁸ It is chosen that with minimal area.

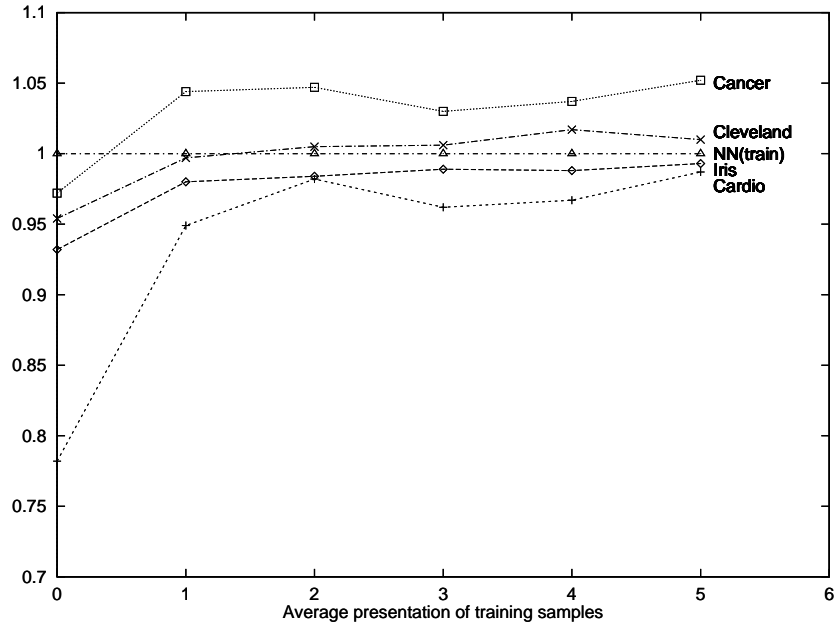


Fig. 2. Ratios of the accuracies of AASM and NN, with increasing average number of time each case is presented for training

improve the accuracy if the reward factor is gradually decreased after a first phase of learning, and the punishment is the only one to act in the second phase. Other experiments were also conducted, which cannot be shown here for lack of space, changing the cardinality of the initial set of seeds. The result is that accuracy increases with the cardinality of the set of initial seeds reaching an asymptotic value. So, applying these techniques one has to balance response times and accuracy choosing an appropriate number of initial seeds.

6 Conclusions and Future Directions

This paper presents a novel approach for learning a non symmetric local metric. The learning algorithm introduced here is based on a variation of the reinforcement/punishment paradigm. A set of experiments performed on standard datasets has shown that good data-compression can be achieved and therefore also good speed-up of run time (query retrieval) performance is obtained.

In the model presented in this paper the learning process starts with an initial set of stored instances (seeds). No method either for selecting a good initial set of seeds or for changing dynamically this initial set, as the training phase proceeds was discussed here. Moving the seeds in the input space would resemble other learning methods, for instance Learning Vector Quantization [13]. Significant improvements of our learning model may be attained as well by introducing a

mechanism that dynamically changes the seeds, that is for deleting stored cases or adding new ones. That would provide a real incremental learning method with a capability to adapt to severe changes in the input space. Another improvement in the accuracy is expected when using a method for dynamically changing the punishment and reward parameters.

An application of the proposed techniques is ongoing on a real application domain whose main goal is to provide support for planning an initial attack to forest fires [21, 7, 20]. We are now acquiring a large case base that is being developed during simulated sessions with a domain expert. Among other advantages, the proposed technique for the automatic adaptation of the similarity metric will add great flexibility to the demonstrator and will simplify the porting of the demonstrator in a different context, for example for operating in a new operational region.

7 Acknowledgements

We would like to thank our anonymous reviewers for their insightful suggestions and remarks. Special thanks to David Aha for helpful discussions and encouragement in pursuing this research. This paper benefited from the editing help provided by Susan Zorat. This work has been partially supported by the EspritIII project #6095 CHARADE (Combining Human Assessment and Reasoning Aids for Decision Making in environmental Emergencies). The partners of CHARADE are Alenia (prime contractor, Italy), Italsoft (Italy), Alcatel ISR (France), Inisel (Spain), ITC-IRST (Italy), Thomson-CSF/SDC and Thomson-CFS/LER (France).

References

1. D. W. Aha. Incremental, instance-based learning of independent and graded concept description. In *Proceedings of the Sixth International Workshop on Machine Learning*, Ithaca, NY, 1989. Morgan Kaufmann.
2. D. W. Aha. A study of instance-based algorithms for supervised learning tasks: Mathematical, empirical and psychological evaluations. Technical Report TR-90-42, University of California, Irvine, 1990.
3. D. W. Aha. Case-based learning algorithms. In *Proceedings of the 1991 DARPA Case-Based Reasoning Workshop*, pages 147–158. Morgan Kaufmann, 1991.
4. D. W. Aha and R. L. Goldstone. Learning attribute relevance in context in instance-based learning algorithms. In *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, pages 141–148, Cambridge, MA, 1990. Lawrence Erlbaum.
5. D. W. Aha and R. L. Goldstone. Concept learning and flexible weighting. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*, pages 534–539, Bloomington, IN, 1992. Lawrence Erlbaum.
6. K. D. Ashley. Assessing similarities among cases: a position paper. In *Proceedings of a Workshop on Case-Based Reasoning*, pages 72–76, Pensacola Beach, FL, 1989. Morgan Kaufmann.

7. P. Avesani, A. Perini, and F. Ricci. Combining CBR and constraint reasoning in planning forest fire fighting. In *Proceedings of the first european workshop on Case-Based reasoning*, pages 235–239, Kaiserslautern, 1993.
8. S. Cost and S. Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10:57–78, 1993.
9. B. V. Dasarathy, editor. *Nearest neighbour (NN) norms: NN pattern classification techniques*. IEEE Computer Society Press, Los Alamitos, CA, 1991.
10. R. Detrano, A. Janosi, W. Steinbrunn, M. Pfisterer, K. Schmid, S. Sandhu, K. Guppy, S. Lee, and V. Froelicher. Rapid searches for complex patterns in biological molecules. *American Journal of Cardiology*, 64:304–310, 1989.
11. P. Domingos. Rule induction and instance-based learning: a unified approach. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence*, 1995.
12. R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.
13. T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, Sept. 1990.
14. M. M. Kokar and S. A. Reveliotis. Reinforcement learning: Architectures and algorithms. *International Journal of Intelligent Systems*, 8:857–894, 1993.
15. D. G. Lowe. Similarity metric learning for a variable-kernel classifier. *Neural Computation*, 7:72–85, 1995.
16. P. M. Murphy and D. W. Aha. *UCI Repository of Machine Learning Databases*. University of California, Department of Information and Computer Science, Irvine, CA, 1994.
17. K. S. Narendra and M. A. Thathachar. *Learning Automata*. Prentice-Hall, 1989.
18. F. Ricci. Constraint reasoning with learning automata. *International Journal of Intelligent Systems*, 9(12):1059–1082, Dec. 1994.
19. F. Ricci and P. Avesani. Learning an asymmetric and anisotropic similarity metric for case-based reasoning. Technical report, IRST, Apr. 1995.
20. F. Ricci, S. Mam, P. Marti, V. Normand, and P. Olmo. CHARADE: a platform for emergencies management systems. Technical Report 9404-07, IRST, 1994.
21. F. Ricci, A. Perini, and P. Avesani. Planning in a complex real domain. In *proceedings of the italian planning workshop*, pages 55–60, Rome, 1993.
22. S. L. Salzberg. A nearest hyperrectangle learning method. *Machine Learning*, 6:251–276, 1991.
23. D. B. Skalak. Representing cases as knowledge sources that apply local similarity metrics. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*, pages 325–330, Bloomington, IN, 1992. Lawrence Erlbaum.
24. C. Stanfill and D. Waltz. Toward memory-based reasoning. *Communication of ACM*, 29:1213–1229, 1986.
25. S. Wess, K.-D. Althoff, and G. Derwand. Using k-d trees to improve the retrieval step in case-based reasoning. In *Topics in Case-Based Reasoning, First European Workshop, EWCBR-93*, pages 167–181, Berlin, 1993. Springer-Verlag.
26. D. Wettschereck. *A study of distance-based machine learning algorithms*. PhD thesis, Oregon State University, 1994.