Optimizing Similarity Assessment in Case-Based Reasoning

Armin Stahl
Image Understanding and Pattern Recognition Group
German Research Center for
Artificial Intelligence (DFKI) GmbH
Technical University of Kaiserslautern
Armin.Stahl@dfki.de

Thomas Gabel
Neuroinformatics Group
Department of Mathematics and Computer Science
Institute of Cognitive Science
University of Osnabrück
thomas.gabel@uos.de

Abstract

The definition of accurate similarity measures is a key issue of every Case-Based Reasoning application. Although some approaches to optimize similarity measures automatically have already been applied, these approaches are not suited for all CBR application domains. On the one hand, they are restricted to classification tasks. On the other hand, they only allow optimization of feature weights. We propose a novel learning approach which addresses both problems, i.e. it is suited for most CBR application domains beyond simple classification and it enables learning of more sophisticated similarity measures.

Introduction

Case-Based Reasoning (CBR) has become a very popular and also commercially successful AI technique. It is based on the assumption that problems can be solved efficiently by reusing knowledge about similar, already solved problems documented in *cases*. In order to solve a new problem, in a first step one has to identify cases which contain the most useful knowledge (Aamodt & Plaza 1994). Since the utility of a case cannot be evaluated directly a-priori, similarity between problem descriptions is used as a heuristic to estimate the cases' expected utility. In general, the quality of this estimation is crucial for the success of any CBR application. Although a lot of CBR applications are based on simple, general applicable distance metrics, many application domains require *knowledge-intensive similarity measures (KISM)* where domain-specific knowledge is used to approximate the cases' utility more accurately (Stahl 2004).

However, since acquiring and encoding this knowledge is a very complex and time consuming task this increases the development costs of a CBR application significantly. Moreover, in many application scenarios the required knowledge is even not available at all during the development phase.

Some early work in the area of instance-based learning (Aha 1991), k-NN classification (Hastie & Tibshirani 1996) and also CBR (Bonzano, Cunningham, & Smyth 1997) has already addressed this problem by applying learning algorithms. However, these approaches are all restricted to simple classification tasks and usually focus on optimizing feature weights (Wettschereck & Aha 1995). On the one hand, CBR is commonly applied for a wide range of application tasks beyond classification, e.g. for building recommender systems in e-commerce or knowledge management scenarios. On the other hand, KISM require more sophisticated representations where feature weights encode only a small part of the domain-specific knowledge. Hence, the existing techniques for learning similarity measures are often not applicable or sufficient in CBR applications.

In order to optimize the similarity assessment in CBR we have proposed a novel learning approach which is based on special feedback about the cases' actual utility (Stahl 2001; Stahl & Gabel 2003; Stahl 2004). By applying accurate learning algorithms, this approach allows optimization of KISM and the incorporation of available background knowledge into the learning process (Gabel & Stahl 2004).

First, the basic idea of our novel learning approach is presented followed by a description of two different learning algorithms that are suited for optimizing feature weights and another important part of KISM, so-called *local similarity measures*. After presenting some evaluation results we conclude with a summary and outlook on future work.

Learning from Relative Utility Feedback

Existing approaches towards learning similarity measures either are based on a statistical analysis of the case base or apply a leave-one-out test to evaluate the accuracy of the retrieval (Wettschereck & Aha 1995). Both approaches are only applicable in classification tasks, since they rely on absolute information about the cases' utility, i.e. a case is either useful because it corresponds to the correct class or not.

However, CBR is applied successfully in many application domains beyond simple classification, e.g. in recommender systems. Here it is mostly difficult to express the actual utility of a case for a given problem in an absolute manner. For example, a customer will have problems expressing the utility of a presented product by using an absolute number such as "0.7". However, in such scenarios the cases' utility can often easily be estimated relatively to other cases. For example, a customer will have no problems to decide that product x is more useful for him than product y.

We have proposed a novel learning approach which allows to exploit such *relative case utility feedback (RCUF)* (Stahl 2005) for optimizing similarity measures (Stahl 2001; 2002; 2004). This approach is based on the existence of

some *similarity teacher* who is able to evaluate retrieval results according to the relative utility of the included cases w.r.t. the given query. As result he will provide a (partially) corrected retrieval result represented by a partial order on (some of) the cases. By defining an *error function* which compares this feedback with the original retrieval result, one obtains a measure for the retrieval quality. This means, the larger the difference between the two partial orders, the higher is the retrieval error. By calculating the average retrieval error for a set of different training queries, one obtains a quality measure for the underlying similarity measure which can be used to start an optimization process. This means, the actual learning task is the search for an optimal similarity measure (corresponding to the minimal retrieval error) in the used representation space.

The advantage of this novel approach is its flexibility. On the one hand, it does not rely on absolute information of the cases' utility which is not available in many application scenarios in contrast to RCUF. On the other hand, it is not restricted to learning feature weights but allows the optimization of arbitrary similarity functions. In the following we assume a typical KISM for computing the similarity between a query Q and a case C[1], consisting of feature weights w_i and feature-specific local similarity measures sim_i:

$$Sim(Q, C) := \sum w_i \cdot sim_i(q_i, c_i)$$

Local similarity measures are commonly represented as *similarity tables* which simply evaluate all pairwise similarity values for symbolic features or *difference-based similarity functions* which map feature differences to similarity values for numeric features (for details see (Stahl 2004)).

In the following we present two different algorithms that are suited for dealing with the described learning task, i.e. the minimization of an accurate retrieval error function given RCUF. While the first algorithm is restricted to learning feature weights, the second algorithm also enables the optimization of local similarity measures.

Optimizing Feature Weights

Like in more traditional similarity measures, feature weights are also a very important part of KISM. So, the definition of accurate weights is a crucial but also very difficult task assuming deep knowledge about the application domain.

We have proposed a gradient descent algorithm for optimizing feature weights on basis of RCUF (Stahl 2001). The algorithm is based on a special error function—called *similarity error*—which is partially differentiable w.r.t. the w_i in order to be able to guide the gradient descent algorithm.

Experimental results have shown that this algorithm is very efficient but also very robust against overfitting (Stahl 2004). Hence a small amount of training data is sufficient for improving the retrieval accuracy of some initial weights (e.g. uniform) significantly.

[1]We assume a simple feature-value based case representation, i.e. $C := (c_1, \ldots, c_n)$

Optimizing Local Similarity Measures

In contrast to optimizing feature weights, there had been no existing approaches for learning knowledge-intensive local similarity measures. For that task, we developed an algorithm that utilizes RCUF and performs search in the space of representable similarity measures using evolutionary algorithms (EA). An EA maintains a population of individuals and evolves it using specialized stochastic operators (crossover and mutation) by which new individuals (offspring) are created. Each individual is associated with a fitness value and the least fit individuals are periodically excluded from the evolution process (selection).

Concerning the learning task faced here, individuals are local similarity measures and the learning algorithm maintains a population for each local similarity measure to be learned. The representation as individuals for similarity tables is done in the straightforward manner by using square matrices. If a difference-based similarity function sim_i is considered, the corresponding individual is represented as a vector of samples of sim_i whose entries are distributed equidistantly over the domain of sim_i as shown in Figure 1.

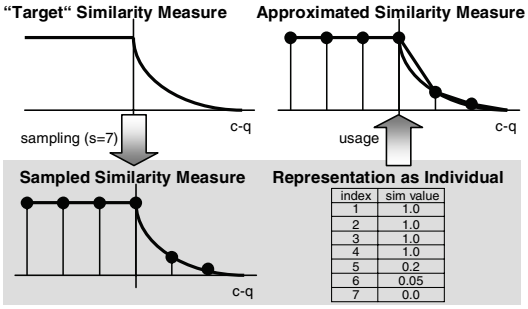

Figure 1: Representation of difference-based similarity functions as individuals to be used within an EA

Given that representation of local similarity measures as individuals within an EA we have designed a number of specialized genetic mutation and crossover operators by means of which—in the course of simulated evolution—new offspring individuals are generated. More details are given by Stahl & Gabel (2003).

Given a specific case attribute and the mandatory training data, our algorithm first settles how to represent the corresponding local similarity measures as individuals. It then proceeds through evolutionary techniques (each genetic operator mentioned is invoked with a certain probability), creates new similarity measures while abolishing old ones, in order to search for the *fittest individual*, whose corresponding similarity measure yields the minimal value of the error function on the training data.

Background Knowledge to Combat Overfitting

In empirical evaluations we have shown that the combination of RCUF and EAs represents a powerful approach to support the modelling of KISM in different application scenarios. The step ahead to also learning local similarity measures, however, leads to a serious problem as well: The

search space to be handled by the learning algorithm is extremely large: In particular, our way of representing local similarity measures by square matrices and vectors of sampled similarity values, allows the EA learner to generate very specific measures. Accordingly, especially when using little training data only, the risk arises that the learner creates models (here, similarity measures) that are overfitted with respect to the training data, while showing poor performance on some independent test data set. Furthermore, the search space is populated with plenty of (local) similarity measures whose usage in practice is extremely implausible. Thus, the EA may waste time searching regions of the search space that do not correspond to meaningful, realistic measures. To tackle these pitfalls we suggested a method to incorporate easily available domain or background knowledge into the learning process, thus guiding and stabilizing it and reducing the danger of overfitting (Gabel & Stahl 2004).

Sources of Knowledge We have identified a number of knowledge sources that may aid the learning process. Roughly, these forms of knowledge can be divided into two groups. On the one hand, *similarity meta knowledge* comprises general demands on the appearance of learned measures. As an example, we defined several basic constraints on the syntactical shape of local similarity measures. One of those concerns the reflexivity of similarity measures: In most application domains a non-reflexive similarity measure would be unpropitious. Another example form of meta knowledge may be derived from the CBR system's case base CB: The case distribution over CB can help to find out which regions of the space of representable similarity measures are worth to be searched thoroughly. It is important to note, that the knowledge acquisition effort for obtaining similarity meta knowledge is comparatively low.

On the other hand, the aid of a knowledge engineer and the incorporation of his *expert knowledge* into the learning process can be highly valuable (in spite of higher knowledge acquisition effort): The human expert may provide a partial KISM definition and may instruct the learner to use this during learning to confine the search space. Then, the remaining, unknown or partially specified parts of the KISM can be learned and fine-tuned using our learning framework.

As a highly desirable secondary effect, this way we have paved the way to enable a hybrid approach to defining KISM. Completely manual definition and fully automated learning of KISM are two extremes of modelling techniques, both featuring certain advantages and drawbacks (Stahl 2002). So, the basic purpose of biasing learning by expert knowledge is to meet in the middle—to use the expert's knowledge on the shape of the similarity measures to be modelled as far as possible and to let the evolutionary learning algorithm determine the remaining parts.

Moreover, we have also proposed a way to utilize the vocabulary knowledge—implicitly contained in the domain model of the respective application domain of the CBR system—as a source to enhance the learning of similarity measures (Gabel, T. 2005).

Interfering the Learning Process To realize the actual restriction of the search space we have introduced the concept of knowledge-based optimization filters (KBOF, Gabel & Stahl (2004)). These are objects that, on the one hand, store the gathered knowledge regarding the learning of similarity measures. On the other hand, they actively interfere the learning process, biasing/directing the search for optimal similarity measures. For the implementation of the evolutionary learning algorithm this implies that a KBOF exerts its influence during offspring creation: Newly generated individuals contradicting too much to the filter's knowledge are discarded. Further, we explicitly allow a KBOF to give advice to the genetic operators adapting their behavior in such a way that more realistic similarity measures are created.

Evaluation Results

A significant advantage of the learning framework we have presented is its applicability to a wide range of application scenarios. Accordingly, we have evaluated the effectiveness of our approach in differing application domains. For example, we focused on a product recommendation scenario and found that learning KISM, that consider the possibilities to customize products (here PCs) in order to ensure the retrieval of adaptable cases, yields substantial improvements in the retrieval of useful cases (Stahl & Gabel 2003). In another product recommendation scenario (used cars) it was our aim to learn similarity measures which reflect customer preferences as accurately as possible. Figure 2 shows the improvement of the recommendation accuracy depending on the result set size when optimizing feature weights by using an increasing amount of RCUF. Detailed results on these experiments are provided by Stahl (2004).

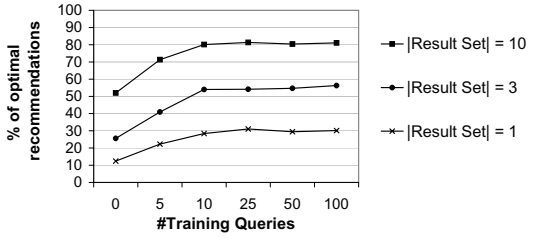

Figure 2: Improvement of recommendation accuracy (measured in percentage of retrieval results that contained the optimal product) in dependency of available amount of RCUF.

To thoroughly examine the effects of incorporating background knowledge into the learning process we have turned to a variety of classification and regression domains available from the UCI Machine Learning Repository[2] and used our framework to learn similarity measures in such a way that the CBR system's prediction accuracy using k-nearest neighbor classification/regression is maximized. Details of the corresponding experiments are given by Gabel & Stahl (2004), some core results are summarized in Figure 3: We consider the classification/regression error a default similarity measure yields (e.g. Euclidean distance) as our baseline

[2] http://www.ics.uci.edu/~mlearn/MLRepository.html

(white bars). Compared to that baseline a KISM learned using our framework produces clearly reduced error rates (dark-gray bars). Here, the results in the top chart are averaged over all application domains we considered, so, error rates relative to the baseline are sketched. Obviously, when learning the similarity measures on the basis of little training data overfitting occurs and only little improvements in prediction accuracy can be achieved. Employing larger training data sets (up to 200), the average classification/regression error can be brought down to about 58% of the error induced by the default measures. In contrast, the lower chart presents the results for one specific classification domain (HayesRoth, 3 classes).

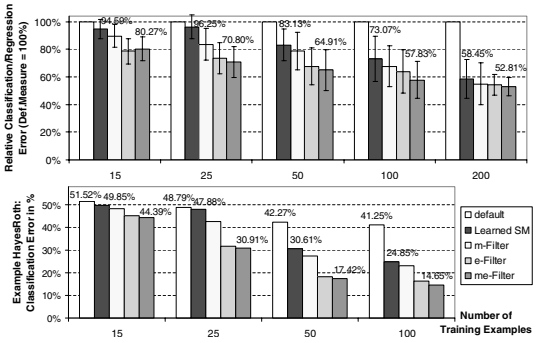

Figure 3: Averaged over six application domains, the learnt similarity measures clearly outperform the default similarity measures that assess similarity based on a syntactic match (top). Exact classification error rates are exemplarily given for the HayesRoth domain (bottom). By incorporating background knowledge into to learning process via KBOFs (m/e/me-Filter) further improvements are possible.

For evaluating the influence of background knowledge on the learning process, we also defined three types of KBOFs for each considered domain: m-Filters contain easily acquirable similarity meta knowledge, e-Filter are enhanced via specific expert knowledge, and me-Filter represent a combination of the former ones, thus constraining the learning process most. The three corresponding data rows in Figure 3 show the improvements in the learning results when optimizing the similarity assessment using our framework *and* guiding the optimization with additional background knowledge. For example, the relative average classification/regression error achieved using 25 training examples in combination with a me-Filter could be decreased to 70.80%, as opposed to the 96.25% error rate achieved without incorporating background knowledge.

Conclusions

In this paper we have presented a novel approach for optimizing the similarity assessment in CBR. The approach avoids the disadvantages of existing learning techniques. On the one hand, it can deal with relative feedback about the cases' utility and hence, it is not restricted to classification tasks. On the other hand, it is the first approach towards learning of knowledge-intensive similarity measures consisting of feature weights and feature-specific local similarity measures. In order to reduce the risk of overfitting, we have proposed additional techniques for incorporating easily to acquire background knowledge into the learning process. Since the presented learning algorithm generates easy understandable representations, the approach may also facilitate the incorporation of explanation approaches. This is an interesting topic for future research.

The results of several experimental evaluations using artificial test domains or UCI data have demonstrated the effectiveness of our approach in different applications scenarios. In a next step we plan to apply our approach also in real world applications. Other interesting issues for future work are the application of alternative learning algorithms or the combination with pure case-based learning techniques.

References

Aamodt, A., and Plaza, E. 1994. Case-based reasoning: Foundational Issues, Methodological Variations, and System Approaches. *AI Communications* 7(1):39–59.

Aha, D. 1991. Case-Based Learning Algorithms. In *Proc. of the DARPA Case-Based Reasoning Workshop*, 147–158. Morgan Kaufmann.

Bonzano, A.; Cunningham, P.; and Smyth, B. 1997. Using Introspective Learning to Improve Retrieval in CBR: A Case Study in Air Traffic Control. In *Proc. of the 2nd International Conference on Case-Based Reasoning*. Springer.

Gabel, T., and Stahl, A. 2004. Exploiting Background Knowledge when Learning Similarity Measures. In *Proc. of the 7th European Conference on Case-Based Reasoning*. Springer.

Gabel, T. 2005. On the Use of Vocabulary Knowledge for Learning Similarity Measures. In *Proc. of the 3rd German Workshop on Experience Management*. Springer.

Hastie, T., and Tibshirani, R. 1996. Discriminant Adaptive Nearest Neighbor Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18(6).

Stahl, A., and Gabel, T. 2003. Using evolution programs to learn local similarity measures. In *Proc. of the 5th International Conference on CBR*. Springer.

Stahl, A. 2001. Learning Feature Weights from Case Order Feedback. In *Proc. of the 4th International Conference on Case-Based Reasoning*. Springer.

Stahl, A. 2002. Defining Similarity Measures: Top-Down vs. Bottom-Up. In *Proc. of the 6th European Conference on Case-Based Reasoning*. Springer.

Stahl, A. 2004. *Learning of Knowledge-Intensive Similarity Measures in Case-Based Reasoning*, volume 986. dissertation.de.

Stahl, A. 2005. Learning Similarity Measures: A Formal View Based on a Generalized CBR Model. In *Proc. of the 6th International Conference on Case-Based Reasoning*. Springer.

Wettschereck, D., and Aha, D. W. 1995. Weighting Features. In *Proc. of the 1st International Conference on Case-Based Reasoning*. Springer.