8th IEEE International Conference on Tools with Artificial Intelligence, Toulouse, France, November 16-19, 1996

Case-Based Classification Using Similarity-Based Retrieval*

Igor Jurisica
Dept. of Computer Science
University of Toronto
Toronto, Ontario M5S 1A4
juris@cs.utoronto.ca

Janice Glasgow
Dept. of Computing and Inf. Sci.
Queen's University
Kingston, Ontario K7L 3N6
janice@qucis.queensu.ca

Abstract

Classification involves associating instances with particular classes by maximizing intra-class similarities and minimizing inter-class similarities. The paper presents a novel approach to case-based classification. The algorithm is based on a notion of similarity assessment and was developed for supporting flexible retrieval of relevant information. Validity of the proposed approach is tested on real world domains, and the system's performance is compared to that of other machine learning algorithms.

1 Introduction

Classification involves associating instances with particular classes; based on the object description, the classification system determines whether a given object belongs to a specified class. In general, this process consists of: first generating a set of categories and then classifying given objects into the created categories. For the purpose of this paper, it is assumed that the categories are known a priori from a prescribed domain theory [38].

Various reasoning techniques have been utilized for the classification task, including neural networks [37], genetic algorithms [12], inductive and instance-based learning [1, 28, 37, 40, 36] and case-based reasoning [31, 4, 15]. Individual approaches are compared to each other based on the method they deploy, the accuracy they achieve, and the complexity of the used algorithm. Most of the systems extract classification rules from training examples during a learning process. Then, they use these rules for classification of unseen instances. Case-based systems, however, store whole cases during the learning process and assess the similarity between a given problem and a stored case in a case base, to determine an appropriate class for a problem.

Evaluating a classifier's performance is not a straightforward process. Individual systems are usually tested on different problem domains, and because of differences in domain complexities, obtained performance measures cannot be compared directly. While performance is highly domain dependent, it is possible to derive evaluation techniques which allow for a meaningful performance comparison of different algorithms [2, 28].

This paper presents a novel case-based classification scheme and evaluates it on several real world domains. The algorithm is based on a notion of relevance assessment [16]. Its flexibility is accomplished by using context in similarity-based retrieval, deploying `Telos` [29] representation language which treats objects and attributes uniformly.

The remainder of the paper is organized as follows: Section 2 describes the classification problem and presents various approaches to the problem. Section 3 introduces the approach to context-based relevance used to define similarity-based retrieval and case-based classification. Context manipulation is used for controlling the classification accuracy. Section 4 presents an evaluation of our system and shows how controlling the relevance of retrieved cases can affect classification accuracy. Section 5 presents concluding remarks.

2 Classification

2.1 Classification Problem Definition

In a classification system, there is usually a fixed set of classes with which a given object can be associated. Classification systems are generally trained

*This research was supported by the Information Technology Research Center of Ontario. Authors are indebted to J. Mylopoulos for helpful comments and suggestions.

first, i.e., already classified objects are presented to them, and the system induces knowledge from these examples (e.g., neural networks change connection weights, decision tree algorithms generate rules and case-based systems remember individual instances). The goal is to learn a correct classification for given examples. Obviously, the system must be able to generalize from these examples, so even unseen examples would be classified correctly and with satisfactory accuracy.

Generally, objects are represented as a collection of properties – attributes or features. In real world domains, objects may or may not be represented properly and/or error-free. Some attribute values might be missing or there may be irrelevant attributes present and relevant attributes missing. Such domains are called imperfect.

In a conventional data analysis, objects are clustered into classes based on a distance (similarity) measure [38]. The similarity between two objects is represented as a number – the value of a similarity function applied to symbolic descriptions of objects. Thus, the similarity measure is context free. Note that such methods fail to capture the properties of a cluster as a whole and are not derivable from properties of individual entities [38].

Supervised learning algorithms for classification can be described as follows:

> Given a sequence of training examples $< \bar{x}_1, c_1 >, \ldots, < \bar{x}_n, c_n >$, where $< \bar{x}_i, c_i >$ is a pair consisting of an object description $\bar{x}_i$ and a proper classification c_i, the classification system must learn the mapping $classify : X \rightarrow C$, where X is the space of objects descriptions and C is the space of possible classes.

Case-based classification classifies instances based on their similarity to stored cases:

> Given a new problem (a case $\mathcal{C}$), the system retrieves set of cases $\mathcal{C}_1, \ldots, \mathcal{C}_k$ from a case base ($\Delta = \{\mathcal{C}_1, \ldots, \mathcal{C}_m\}$) and classifies the new problem based on the retrieved matches.

2.2 Overview of Some Existing Approaches to Classification

Traditional approaches to classification involve generating a set of rules, based on induction from training examples. The requirement for such algorithms is that created rules correctly classify known (i.e., training) examples and perform well on unseen (i.e., test) examples [32].

ID3 [33] is a classification system based on a decision tree algorithm. Using the induction from training examples, ID3 generates a decision tree – a classification rule that examines the values of some attributes of an object in order to assign it into a proper class.

AC^3 [38] is a classification algorithm based on attribute-based conjunctive conceptual clustering – a way of grouping objects into conceptually simple classes. Using this method, a set of objects forms a class only if it can be described by a conjunctive concept involving relations on object attributes.

AUTOCLASS II [9] is an induction algorithm used to discover classes from databases based on Bayesian statistical techniques. The system determines the number of classes, their probabilistic descriptions and the probability that each object is a member of a given class. This allows for making each and every attribute potentially significant as well as assigning objects to different classes. The system also allows for identifying hierarchies of attributes, selecting common attributes and distinguishing attributes between classes.

IB1 [2, 5] is a nearest-neighbor, instance-based learning system used for classification and for control tasks. Based on a problem description, IB1 retrieves the k-nearest neighbors and uses them to supply a solution to a given problem. Since all attributes are used during retrieval, the system's performance decreases quickly with the number of irrelevant attributes. Performance degradation of IB1 with the number of irrelevant attributes can be solved either by data pre-processing (removing all irrelevant attributes) or by the use of an intelligent, selective partial matching algorithm. The latter approach is similar to m_of_n concepts in machine learning [30], i.e., the system considers only $m < n$ attributes during retrieval. There are various methods used to decide m and Ortega [30] presents an evaluation of system performance for various m. In addition to deciding the size of m, the approach described in this paper allows for specifying which attributes to exclude and for posing additional constraints on attribute values, if desired.

CB1 [15] is a PAC (Probably Approximately Correct) learning, case-based classification algorithm designed for general purpose learning. Even though it is relatively inefficient, considering the sample complexity, it is an interesting step towards a computational learning theory for CBR systems.

PROTOS [31] is a a case-based system used for

heuristic classification in the clinical audiology domain. The system uses exemplar-based learning [26, 6, 7, 3], a method especially appropriate for domains lacking a strong domain theory. In the system, classification is combined with explanation. Explanation is used for justifying case classification and for determining similarity between two cases.

An incremental retrieval system, I-CBR, is presented in [10]. For this case-based reasoning paradigm, the user specifies a case skeleton and attempts to retrieve all cases similar to it. I-CBR partitions case attributes into two groups: the ones which are known immediately and the ones that need some extra effort for specifying them. During the retrieval process, the user specifies known attributes and the system returns a pool of possible case candidates. From the set of unknown attributes the most discriminating one is selected and the user is queried for its value. Thus, the initial pool of retrieved cases is processed to decrease its size to eliminate less-similar cases which increases classification accuracy.

The most appealing feature of the similarity-based retrieval approach proposed in this paper is that it allows for retrieving relevant information even without a complete and precise query or a perfect match. Such retrieval is an essential part of a case-based classification system. After retrieving relevant (similar) cases, the case-based classification system can adapt previous solutions and use them to predict the class for a current case.

3 The $\mathcal{TA}3$ Case-Based Classification System

In this section we present the theory behind case-based classification, using flexible relevance assessment [16, 17, 20, 19, 22] – a fundamental component of the $\mathcal{TA}3$ case-based reasoning system. We define relevance in terms of context and similarity, and we also show how context can be used to control the accuracy of the classification process.

Although it has been acknowledged that context plays a central role in determining similarity, previous works on similarity measures for case-based reasoning generally assume that context is implicit in the case representation, or is acquired through machine learning techniques [8]. We define context as a parameter of a similarity relation and demonstrate the monotonicity of case retrieval.

A case, $\mathcal{C}$, corresponds to a real world situation, represented as a structured object with relations [29]. Objects and relations are represented as a set of attributes and their values, denoted $\mathcal{C} = \{< a_0 : V_0 >, \cdots, < a_n : V_n >\}$, where $< a_i : V_i >$ is an attribute/value pair. To explain the theory, we will use a running example from the servo domain, presented in Section 4.2. Here, the case is a collection of five attributes with values:

```
Example 1
              MOTOR:  E
              SCREW:  D
              PGAIN:  4
              VGAIN:  5
         RISE_TIME:  0.28125095
```

Using the information about the usefulness of individual attributes and its properties, attributes are grouped into one or more Telos-style categories [29]. In a servo domain (see Section 4.2), three categories are used: *motor* and *screw* form the first category, *pgain* and *vgain* comprise the second category, and the last category is the class – *rise time*. The membership is defined either using domain knowledge (if available) or using a knowledge discovery algorithm [11, 19, 24]. This grouping allows for ascribing different constraints to different groups of attributes, and the process of retrieving relevant cases can be described as a constraint satisfaction process [39, 13, 27]. Categories allow for improved system performance, as will be shown later.

An explicitly specified context allows for less strict matching than equivalence. The goal is to retrieve not only exact matches (equivalent cases), but partial matches (similar cases) as well. In short, context is a parameter of a relevance relation which maps a case base onto a set of relevant (in terms of context) cases. A context is defined as a finite set of attributes with associated constraints on the attribute values, denoted $\Omega = \{< a_0 : CV_0 >, ..., < a_k : CV_k >\} = \{< a_i : CV_i >\}_0^k$, where a_i is an attribute name and constraint CV_i specifies the set of "allowable" values for attribute a_i. A possible context in a servo domain can be defined as follows:

```
Example 2
              MOTOR:  D or E
              SCREW:  C or D or E
              PGAIN:  4 or 5
              VGAIN:  5
         RISE_TIME:  any value
```

Thus, a context allows for controlling what can and what cannot be considered as a partial match. As a result, context controls the classification accuracy in case-based classification, and the classification will

not degrade with irrelevant attributes present. In addition, by modifying the context, the system may flexibly change the tradeoff between accuracy and speed [20].

For the purpose of performance evaluation in terms of precision and recall, we define relevance using context and similarity. There is little commonality between definitions of relevance used in different systems [14]. The most common ground is reached when relevance is defined as something useful. Our definition of relevance is based on the assumption that the more similar in a given context the problem description is to the case in a case base, the higher the accuracy for case-based classification, and thus the more relevant the case.

In the proposed theory, similarity is considered as a relation between cases. It is defined to supplement equivalence by allowing for partial matches. Cases are considered similar if they satisfy a given context. We say that a case $\mathcal{C}$ **satisfies** (or matches) a particular context Ω, denoted $sat(\mathcal{C}, \Omega)$, if and only if for each attribute a_i specified in the context, the value V_i of that attribute in the case satisfies the constraint (CV_i), specified by the context:

$$sat(\mathcal{C}, \Omega) \; iff \; \forall(< a_i : CV_i > \in \Omega \; \rightarrow$$
$$\exists(< a_i : V_i > \in \mathcal{C} \; \wedge \; V_i \in CV_i)).$$

Using our running example, *motor* attribute is satisfied since the value E is part of the constrain defined in the context ($D \; or \; E$). Other attributes are processed similarly.

Since the context affects the class into which the input problem is assigned, context specification is vital for the correct answer, i.e., for the accurate classification. In general, the context can be specified using the following scenarios:

1. The user has enough domain knowledge to specify the context (query-by-example can be used as an initial context specification).

2. The user specifies the task (s)he wants to solve and the system selects an appropriate context (task-based retrieval).

3. First, a knowledge discovery algorithm is used to locate relevant attributes and attribute values. Then, a user forms a context for retrieval using this information. This approach is suitable for novice users.

4. The user submits an ad hoc query, reviews the resulting solution, and then iteratively modifies and resubmits the query (retrieval by reformulation). This approach is best suited for repository browsing.

Using the context, the input problem and cases in a case base are interpreted and their similarity is assessed. A case $\mathcal{C}_1$ is **similar** to a case $\mathcal{C}_2$ for a given context Ω, denoted: $\mathcal{C}_1 \sim_\Omega \mathcal{C}_2$, if and only if both $\mathcal{C}_1$ and $\mathcal{C}_2$ satisfy context Ω:

$$\mathcal{C}_1 \sim_\Omega \mathcal{C}_2 \; \leftrightarrow \; sat(\mathcal{C}_1, \Omega) \wedge sat(\mathcal{C}_2, \Omega).$$

Using the case and the context defined in Example 1 and in Example 2, and the following case:

```
Example 3
              MOTOR:  C
              SCREW:  C
              PGAIN:  4
              VGAIN:  5
         RISE_TIME:  0.32456011
```

we see that the two cases are not similar in a given context because the *motor* attribute value of the second case does not satisfy the constraint. If the *motor* value is changed from C to D (or the context is changed to *motor:* $C \; or \; D \; or \; E$) only then the two cases would be similar in a given context.

For the purpose of case retrieval, the similarity measure can be viewed as a relation which maps a context Ω and a case base Δ onto the set of cases SI in the case base that satisfy the context, i.e., $retrieve : \Omega \times \Delta \rightarrow SI$. Because all cases in the set SI satisfy the context, they are similar to each other in a given context Ω. Following the idea behind case-based reasoning, all cases in SI are considered relevant in a given situation, specified by the context Ω. Thus, in case-based classification, the classification for all cases relevant in a given context is the same class. The complexity of this task is characterized by the number of required comparisons which is, in the worst case, $\mid \Omega \mid \times \mid \Delta \mid$. Case base organization (e.g., clustering or indexing) may improve the efficiency of performing the task by limiting the search space of the case base.

Using the presented theory, we define a case-based classification algorithm (CBC) as part of the $\mathcal{TA}$**3** case-based classification system as follows:

Input: A case base Δ, a context Ω and a problem $\mathcal{C}_p$ (also called an input case, i.e., a case without a solution or appropriate class).

Output: A class c_k for a problem C_p ($< \mathcal{C}_p, c_k >$):

$$classify(\mathcal{C}_p) \rightarrow < \mathcal{C}_p, c_k > \leftrightarrow$$
$$\exists \mathcal{C}_r \in \Delta \; : \mathcal{C}_p \sim_\Omega \mathcal{C}_r \wedge \; < \mathcal{C}_r, c_k > .$$

Using the definition of similarity ($\sim_\Omega$) and CBC, it is easy to see that by manipulating the context (a form of bias) it is possible to control the quantity and quality of retrieved cases. Since CBC is based on the idea that similar past cases determine the class for a given problem, context manipulation can be used to change the classification for the problem. It should be noted that the context cannot be arbitrarily chosen. For a correct answer, it must include attributes and constraints that would allow for organizing cases into clusters based on classification, i.e., it must include predictive attributes.

During the retrieval, there are 3 possible results:

1. No relevant cases are retrieved – the system cannot classify a given problem.

2. All relevant cases belong to the same class – the input problem is assigned to the class suggested by retrieved cases.

3. Retrieved cases belong to different classes – the system assesses the relevance of individual cases and assigns the input problem into the most probable class. If this is not possible, the majority class is selected. If this step fails as well, then the system returns all possible classifications with an "I do not know" answer.

Context Relaxation and Restriction. Context relaxation allows for the retrieval of more cases from the case base. If we relax the context – the matching criteria become less tight – more cases will match (in the extreme, if we specify no constraints in context, all cases from a case base would match).

1. **Cardinality Relaxation.** The satisfiability definition can be relaxed so that a case, represented by n attributes, may satisfy a context if it matches the context for some specified number $m < n$ attributes, rather than requiring all the attribute constraints in the context be satisfied [30]. We call this cardinality relaxation.

   Consider the example presented in Example 3. Using cardinality relaxation for the first category (*motor* and *screw*), we may change the requirement that both attributes must satisfy the constraint to the requirement that if either of the two attributes satisfies the requirement then the whole category would satisfy it.

2. **Value Relaxation.** The range of values can be expanded to include semantically neighboring values, i.e., attribute value are relaxed. In order to allow for controlled relaxation, attributes may have an attached priority. Then, during the cardinality context modifications, the system modifies attributes with the lowest priority. If no priority is explicitly assigned to attributes, then the first attribute is considered as the most important.

   Considering the example presented in Example 2, the two cases would match if the constraint on the *motor* attribute is relaxed to include value C, i.e., *motor: C or D or E*.

Similar to relaxation, contexts can be iteratively **restricted** to retrieve successively fewer cases. Restricting a context means making it more specific. A restriction function takes a context, restricts it according to criteria and returns a modified context. Either cardinality or value restriction can be applied, depending on criteria. During restriction, the category with the lowest priority is restricted first. If not successful, the cardinality with the second lowest priority is used, etc. Only one category is restricted at a time.

A context Ω_1 is a **relaxation** of a context Ω_2, denoted $\Omega_1 \succeq \Omega_2$, if and only if for all pairs $< a_i, CV_i > \in \Omega_1$, if there exists a pair $< a_j, CV_j > \in \Omega_2$ such that if $a_i = a_j$ then $CV_i \supseteq CV_j$. Conversely, if Ω_1 is a relaxation of Ω_2 then we say that Ω_2 is a **restriction** of Ω_1, denoted $\Omega_2 \preceq \Omega_1$.

$$\Omega_1 \succeq \Omega_2 \; iff \; \forall < a_i : CV_i > \in \Omega_1,$$

$$\exists < a_j : CV_j > \in \Omega_2 \;\; : \;\; a_i = a_j \wedge CV_i \supseteq CV_j.$$

Context relaxation and restriction is used during retrieval to control the quantity and quality (closeness) of cases considered relevant. Thus, modifying the context, the system may return an approximate answer quickly or may spend more resources getting an accurate answer. An approximate answer can be iteratively improved, so that the change between approximate and accurate answer is continuous.

4 Performance Evaluation

Performance evaluation must be conducted with a special care on the used data set and on the selected measures [2]. It is not only important to know what to measure when evaluating a system but also to know how to interpret the results. There are many accepted benchmarks used in different fields and there are numerous evaluations available. Yet,

many times wrong conclusions are being made or even worse, useless measures are being considered.

System performance should be determined with respect to *time* and *task*. The first characteristic reveals if the system can perform the specified task (e.g., retrieve all relevant cases from the case base). Usually, task-performance of a classification system is measured by evaluating its *accuracy*, i.e., percentage of correct classifications. The second characteristic measures how long it takes for the system to perform the specified task. In addition, scalability measures task/time performance dependability on the case base size.

It should be noted, that accuracy is not only system dependent but is also strongly domain dependent [2]. In [28] a fair evaluation criterion is suggested to measure classification accuracy. The motivation for the study is that a simple comparison of classification accuracy might be misleading. For example, 80% accuracy in a perfect domain with 2 classes can be achieved trivially whereas 40% accuracy in a domain with 26 classes (and possible missing information) might be hard to achieve.

Based on the proposed theory, we have applied the research prototype $\mathcal{TA}$**3** to several real world domains:[1] a medical domain ($\mathcal{TA}$**3**$_{IVF}$) [25, 24], a servo-mechanism ($\mathcal{TA}$**3**$_{Servo}$) and robotic domain ($\mathcal{TA}$**3**$_{Robot}$) [21, 22], a letter classification domain ($\mathcal{TA}$**3**$_{Letter}$) [19] and a small software case base ($\mathcal{TA}$**3**$_{SR}$) [18]. Each domain has different characteristics and thus allows for a better performance evaluation. Retrieved cases were used for predicting unknown values for attributes. Thus, simple cross-validation (leave-one-out method) was used and results were compared to actual cases to avoid subjective bias in assessing their relevance. Prediction is successful only if relevant cases are used as a starting point. As a result, the accuracy of classification (i.e., percentage of correctly classified problems) measures the retrieval capability of $\mathcal{TA}$**3**.

The main objective of our evaluation was to test task performance. Time performance is reported elsewhere due to the lack of space [23].

4.1 Problem 1: IVF Case Base

In-vitro fertilization (IVF) domain involves medical records about patients. The case base which is available to us consists of 788 cases and 55 attributes per case (after confidential information has been removed). Out of 788 cases, 149 are clinically successful pregnancies, 10 are pregnancies with ovarian hyper-stimulation complication, 12 are pregnancies ended by abortion, 4 are ectopic pregnancies and 632 are unsuccessful pregnancies (there is not information about pregnancy for 7 cases).

The two most common tasks performed by physicians are classification and knowledge mining. Here we report on using $\mathcal{TA}$**3** to support the classification task, which can be described as follows:

1. Predicting a Treatment. Having initial information about the patient (age, diagnosis of infertility, previous treatment history, etc.), the task is to retrieve similar patients from the case base and to suggest such a treatment for the current patient that the probability of a successful pregnancy is increased. This involves suggesting the day of human chorionic gonadotrophin administration (DAY_HCG) and the number of ampoules of human menopausal gonadotrophin (NO_HMG). After retrieving similar cases, cases with pregnancy are considered as successful examples and cases without pregnancy as a negative cases. DAY_HCG and NO_HMG serve as classes and since both have continuous values, this constitutes classification into a continuous class, i.e., classification into an infinite number of possible classes instead of classification into a finite number of discrete classes.

2. Predicting Pregnancy Outcome. After the initial treatment is completed, additional attributes are available (e.g., response to the current treatment). The task is to predict the outcome of the whole treatment, i.e., to predict pregnancy outcome. This involves determining values for the following attributes: pregnancy, abortion, ovarian hyper-stimulation syndrome and ectopic pregnancy. This process can be described as a classification into discrete classes with possible values "yes" or "no".

In order to statistically evaluate classification, we have conducted a series of tests. Using a simple cross-validation (leave-one-out method and random case selection with results averaged over 20 trials), we obtained the results presented in Table 1. For predicting pregnancy outcome, the system was able to achieve 60.6% accuracy using only the first eight attributes and 71.2% accuracy using a series of estrogen values in addition (15 more attributes).

4.2 Problem 2: Servo Database

The servo database[2] was previously used for testing various machine learning algorithms [34, 35].

[1] $\mathcal{TA}$**3**$_X$ denotes application of $\mathcal{TA}$**3** into a domain X.

[2] Data description and performance results obtained by other machine learning system are available at ftp://ics.uci.edu/pub/machine-learning-databases/servo/

The database covers a non-linear phenomenon – predicting the rise time of a servo-mechanism in terms of two (continuous) gain settings and two (discrete) choices of mechanical linkages. Thus, the rise time is the class with infinite number of values.

The data set consists of 167 instances with 5 attributes: *motor, screw, pgain, vgain* and the class *rise time*. All attributes in the case are divided into three Telos-style categories: category 1 – *motor* and *screw*, category 2 – *pgain* and *vgain*, and category 3 – *rise time*. Even though this domain is neither complex nor large, it is of interest to us since several machine learning algorithms have been evaluated and compared with respect to it. Table 2 summarizes results for $\mathcal{TA}3_{Servo}$ and other machine learning techniques, obtained by using simple cross-validation (leave-one-out method) and averaged over 10 and 60 random trials respectively.

$\mathcal{TA}3_{Servo\,1}$ used value relaxation where only the *motor* attribute was relaxed and only to the left side (e.g., value B would be relaxed to A *or* B). $\mathcal{TA}3_{Servo\,2}$ results were obtained by relaxing value only for the *motor* attribute but in both directions (e.g., value B would be relaxed to A *or* B *or* C). It should be noted that relaxing A and E in both scenarios would be the same – A or B and D or E. In a situations where the initial relaxation did not yield a result, additional methods are used. First, cardinality relaxation for *motor* and *screw* is tried, i.e., all cases that match *motor*, *screw* or both are retrieved. Second, cardinality relaxation for *pgain* and *vgain* is applied.

Even though (direct) error comparison can be used for performance evaluation and system comparison, it is not sufficient in general. In the data presented in Table 2 there is no information on the confidence in the results, nor is there any information about the significance of the errors or about the confidence intervals. In other words, statistical evidence is missing. Thus, we have also conducted a statistical evaluation of our system. The results show that it is advantageous to have an option to change the relaxation technique and also that the servo domain has better informativeness then the IVF domain since higher accuracy can be achieved with the same system.

Attribute	*AAE*	*RE*
NO_HMG	3.2 ± 1.98	0.15 ± 0.09
DAY_HCG	0.9 ± 0.558	0.07 ± 0.04

Table 1. IVF domain – Statistical results with 95% confidence level. Absolute and relative errors (AAE, RE) between suggested and actual values for the treatment, namely day of human chorionic gonadotrophin administration (DAY_HCG) and the number of ampoules of human menopausal gonadotrophin (NO_HMG).

Method	*AAE*	*RE*
Guessing mean	1.15	1.00
Instance-based	0.52	0.26
Regression	0.86	0.49
Model trees	0.45	0.29
Neural nets	0.30	0.11
Regression+IBL	0.48	0.20
Model trees+IBL	0.30	0.17
NN+IBL	0.29	0.11
$\mathcal{TA}3_{Servo\,1}$	0.118 ± 0.042	0.073 ± 0.032
$\mathcal{TA}3_{Servo\,2}$	0.09 ± 0.035	0.056 ± 0.012

Table 2. Absolute and relative errors (AAE, RE) on servo-data in terms of a difference between actual and computed value of the rise time of the servo-mechanism (IBL - instance-based learning). $\mathcal{TA}3_{Servo}$ results are presented for 60 random trials, with 95% confidence levels.

4.3 Problem 3: Robotic Domain

A non-linear phenomenon – predicting the joint angles (φ_1, φ_2 and φ_3), for the three-link spherical angular robot when given desired end-effector coordinates is covered in the robotic domain. This task can be characterized as a classification into a continuous class, namely classification into an infinite number of real values (the triplet φ_1, φ_2 and φ_3).

The task under consideration is known as an inverse kinematic, e.g., computation of the robot's joint angles to ensure that the end-effector coordinates will reach a desired position. This is a complex task which might be computationally intractable for complex systems. In such situations, other methods are used to approximate the solution. In our study we used an existing small robot as an example. Robot's parameters are presented in [21].

One obvious need for the inverse kinematic task is during planning – a robot's end-effector is at point

X and should be moved to point Y. For example, in welding the usual task is to follow a particular curve during the process. Thus, the inverse kinematic must be computed reasonably fast.

The presented results are for a uniformly generated database which consists of 2,000 instances, where each instance is represented by 9 attributes containing real values: three lengths describing robotic arms, three attributes describing a desired end-effector position and three joint angles. Individual attributes are grouped into three Telos-style categories: robot's parameters, end-effector position and joint angles.

During evaluation we used the leave-one-out method and we computed significance intervals and variances from the average values over 20 random trials (Table 3). The uniform generation of cases is especially useful for off-line learning to guarantee a reasonable answer in any possible situation. Another possibility is to "observe" actual cases while the robot is performing normal tasks. The problem with this approach is that it requires on-line operation and that the working space may not be represented evenly within the case base. Thus, the system will be able to work in similar situations to the ones observed during learning, but the performance will degrade if there is a change in the robot task. On-line learning can be a useful experience added to the off-line generate knowledge base, which can be used for generalizations and for knowledge-based adaptation of retrieved cases.

During the experiment, different criteria for selecting relevant cases have been tested. In general, case representation was organized into three Telos-style categories, one of which contained the class. Only cardinality relaxation was used in $\mathcal{TA}3_{Robot\ 1}$ (i.e., for two categories, only m_of_3, $1 < m < 3$, attributes were required to match). $\mathcal{TA}3_{Robot\ 2}$ used attribute value relaxation first, and only when this failed was cardinality relaxation performed.

4.4 Problem 4: Character Recognition Domain

The letter classification task has previously been used for machine learning algorithm testing [12, 2].[3] The data set consists of 20,000 instances described by 17 attributes: letter, horizontal and vertical position of a box, width and height of a box, total number of pixels, etc. All attributes are grouped into five different categories. The objective is to classify each of a large number of black-and-white rectangular pixel displays as one of the 26 capital letters in the English alphabet. The character images were based on 20 different fonts and each letter within these 20 fonts was randomly distorted to produce a database. Each stimulus was converted into 16 primitive numerical attributes (statistical moments and edge counts) which were then scaled to fit into a range of integer values from 0 through 15.

Method	*Accuracy A*	*Accuracy B*
Backprop.	—	81.9 ± 0.6%
IB1	95.7 ± 0.4%	81.7 ± 0.7%
CN2	87.9 ± 0.8%	68.7 ± 1.0%
C4	86.4 ± 0.7%	67.4 ± 0.8%
Genetic Alg.	82.7	—
$\mathcal{TA}3_{Letter\ 1}$	90% ± 0.1461	80% ± 0.2011
$\mathcal{TA}3_{Letter\ 2}$	100%	85% ± 0.1742
$\mathcal{TA}3_{Letter\ 3}$	100%	95% ± 0.1001
$\mathcal{TA}3_{Letter\ 4}$	100%	100%

Table 4. Character recognition domain – Statistical results with 95% confidence level. Percentage of correct letter classifications over 20 random trials using 20,000 and 2,000 cases (accuracy A and B).

For the classification task the system is presented with 16 attributes describing a letter and the task is to classify the letter, based on previously seen cases. Performance evaluation, based on simple cross-validation is summarized in Table 4. All algorithms but $\mathcal{TA}3_{Letter}$ were trained on the first 16,000 and 1,600 cases respectively and then tested on the remaining 4,000 (and 400) cases. Reported results are the average values over 10 trials. Since $\mathcal{TA}3_{Letter}$ does not need a learning period, we used the leave-one-out testing method on the first 2,000 and on all 20,000 cases respectively. Results are the average values over 20 random trials.

In $\mathcal{TA}3_{Letter\ 1}$ we used a modified nearest-neighbor approach – attributes were grouped into categories for selective cardinality relaxation/restriction. $\mathcal{TA}3_{Letter\ 2}$ was obtained by using a value relaxation (i.e., relaxing attribute values to include values of immediate neighbors). If more cases were needed, cardinality relaxation was performed. $\mathcal{TA}3_{Letter\ 3}$ was obtained by using a combination of previous approaches with equal voting for each of them. $\mathcal{TA}3_{Letter\ 4}$ was obtained by using different category groupings for attributes. In the previous test, categories were created sequentially, each group having four attributes (letter attribute

[3] Data description and performance results obtained by other machine learning system are available at ftp://ics.uci.edu/pub/machine-learning-databases/letter-recognition/

Method	*Average Abs. Error*			*Relative Error*		
	φ_1	φ_2	φ_3	φ_1	φ_2	φ_3
$\mathcal{TA}3_{Robot\ 1}$	0.4855 ± 0.2128	0.0	0.0	0.00586863 ± 0.00257	0.0	0.0
$\mathcal{TA}3_{Robot\ 2}$	0.0	2.5 ± 1.0957	0.0	0.0	0.0274725 ± 0.0120	0.0

Table 3. Robotic domain – Statistical results with 95% confidence level. Absolute and relative errors between actual and computed values for three joint angles.

was an extra category). In the latter test we used a knowledge mining tool to find out the most relevant attributes in recognizing individual letters and we grouped attributes accordingly.

5 Conclusions

Similarity-based retrieval tools can advantageously be used in building flexible retrieval and classification systems. Case-based classification uses previously classified instances to classify unknown instances. Classification accuracy is affected by the retrieval process – the more relevant the instances used for classification, the higher the accuracy. Using different context and criteria affects the time spent on searching for the solution as well as the quality of the solution. $\mathcal{TA}3$ allows for tuning up the reasoning to meet different requirements, similarly as in [10]. On the one hand, in the medical domain [25, 24], a more accurate suggestion for the hormonal therapy has a positive impact on pregnancy and is also cost effective, since it minimizes the quantity of hormones given to the patient. Thus, even though the treatment should be suggested reasonably fast, clearly accuracy is more important. On the other hand, in the robotic domain [21], time resources are limited. Thus, even an approximate solution provided in real time has a higher value than an accurate solution delivered late. In the inverse kinematic task, there are fast techniques available to produce an accurate solution from an approximate one. However, for some robotic architectures, there might not be a computational solution to the problem. Thus, the main objective is to have a solution available within time constraints.

We briefly introduced the $\mathcal{TA}3$ system, which uses context-based similarity to retrieve relevant cases and then uses them for the classification task. The validity of the proposed approach is tested on real world domains and the performance of the proposed system is compared to the performance of other machine learning algorithms. The main advantage of the context-based approach is that classification does not degrade with irrelevant attributes, since they can be ignored during the retrieval process and thus would not affect the solution, similarly as in [30].

References

[1] D. Aha, D. Kibler, and M. Albert. Instance-based learning algorithms. *Machine Learning*, 6(1):37–66, 1991.

[2] D. W. Aha. Generalizing from case studies: A case study. In *The 9^{th} International Conference on Machine Learning*, pages 1–10, Aberdeen, 1992.

[3] D. W. Aha. An implementation and experiment with the nested generalized exemplars algorithm. Technical Report AIC-95-003, Naval Research Laboratory, Navy Center for Applied Research in Artificial Intelligence, Washington, DC, 1995.

[4] D. W. Aha and R. L. Bankert. Feature selection for case-based classification of cloud types: An empirical comparison. In *AAAI-94 Workshop on Case-Based Reasoning*, Seattle, WA, 1994.

[5] D. W. Aha and S. L. Salzberg. Learning to catch: Applying nearest neighbor algorithms to dynamic control tasks. In P. Cheeseman and R. W. Oldford, editors, *Selecting Models from Data: Artificial Intelligence and Statistics IV*. Springer-Verlag, 1994.

[6] E. Bareiss, B. W. Porter, and C. Wier. The exemplar-based learning apprentice. Technical Report AI87-53, The Univ. of Texas at Austin, 1988.

[7] L. K. Branting. Integrating generalizations with exemplar-based reasoning. In *Proc. of the 11^{th} Annual Conference of the Cognitive Science Society*, pages 139–146, Ann Arbor, MI, 1989.

[8] T. Cain, M. J. Pazzani, and G. Silverstein. Using domain knowledge to influence similarity judgments. In *Proceedings of the Case-Based Reasoning Workshop*, pages 191–198, Washington, DC, 1991.

[9] P. Cheeseman, J. Kelly, M. Self, J. Stutz, W. Taylor, and D. Freeman. Autoclass: A Bayesian classification system. In *Proc. of the 5^{th} International Conference on Machine Learning*, pages 54–64, Ann Arbor, MI, 1988.

[10] P. Cunningham, A. Bonzano, and B. Smyth. An incremental case retrieval mechanism for diagnosis. Technical Report TCD-CS-95-01, Trinity College, Dublin, Ireland, 1995.

[11] J. Frawley and G. Piatetsky-Shapiro. *Knowledge Discovery in Databases*. AAAI Press, 1991.

[12] P. W. Frey and D. J. Slate. Letter recognition using Holland-style adaptive classifiers. *Machine Learning*, 6(2), 1991.

[13] T. Gaasterland. Restricting query relaxation through user constraints. In *Proc. on International Conference on Intelligent and Cooperative Information Systems*, pages 359–366, Rotterdam, 1993.

[14] R. Greiner. *AAAI Fall Symposium Series on Relevance*. AAAI Press, Menlo Park, CA, 1994.

[15] A. D. Griffiths and D. G. Bridge. On concept space and hypothesis space in case-based learning algorithms. In *Proc. of 8th European Conference on Machine Learning*, 1995.

[16] I. Jurisica. How to retrieve relevant information? In *[14]*, pages 101–104, 1994.

[17] I. Jurisica. $\mathcal{TA}$**3** : Case-based intelligent retrieval and advisory tool. In *ACM Conference on Society and the Future of Computing*, Durango, CO, 1995.

[18] I. Jurisica. A similarity-based retrieval tool for software repositories. In *The Third Workshop on AI and Software Engineering: Breaking the Mold. IJCAI-95*, Montreal, Quebec, 1995.

[19] I. Jurisica. Inductive learning and case-based reasoning. In *Canadian AI Conference, Workshop on What is inductive learning?*, Toronto, 1996.

[20] I. Jurisica. Supporting flexibility. a case-based reasoning approach. In *The AAAI Fall Symposium. Flexible Computation in Intelligent Systems: Results, Issues, and Opportunities*, Cambridge, 1996.

[21] I. Jurisica and J. Glasgow. Applying case-based reasoning to control in robotics. In 3rd *Robotics and Knowledge-Based Systems Workshop*, St. Hubert, Quebec, 1995.

[22] I. Jurisica and J. Glasgow. A case-based reasoning approach to learning control. In 5th *International Conference on Data and Knowledge Systems for Manufacturing and Engineering, DKSME-96*, Phoenix, Arizona, 1996.

[23] I. Jurisica, J. Mylopoulos, and J. Glasgow. Performance analysis of an incremental algorithm for the $\mathcal{TA}$**3** CBR system. 1996. Submitted.

[24] I. Jurisica, J. Mylopoulos, J. Glasgow, H. Shapiro, and R. F. Casper. Case-based reasoning in IVF: Prediction and knowledge mining. *AI in Medicine. Special issue on CBR in Medicine*, 1996. Under revisions.

[25] I. Jurisica and H. Shapiro. A computer model for case-based reasoning in IVF. In *The 51st Conference of the American Society for Reproductive Medicine*, Seattle, Washington, 1995.

[26] D. Kibler and D. Aha. Learning representative exemplars of concepts: An initial case study. In *Proc. of the 4th International Workshop on Machine Learning*, pages 24–30, Irvine, CA, 1987.

[27] T. Kokeny. Constraint satisfaction problems with order-sorted domains. *International Journal on Artificial Intelligence Tools*, 4(1 & 2):55–72, 1995.

[28] I. Kononenko and I. Bratko. Information-based evaluation criteria for classifier's performance. *Machine Learning*, 6(1):67–80, 1991.

[29] J. Mylopoulos, A. Borgida, M. Jarke, and M. Koubarakis. Telos: Representing knowledge about information systems. *ACM Transactions on Information Systems*, 8(4):325–362, 1990.

[30] J. Ortega. On the informativeness of the DNA promoter sequences domain theory. *Journal of Artificial Intelligence Research*, 2:361–367, 1995.

[31] B. W. Porter, E. Bareiss, and R.C.Holte. Concept learning and heuristic classification in weak-theory domains. *Artificial Intelligence*, 45:229–263, 1990.

[32] J. R. Quinlan. The effect of noise in concept learning. In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, pages 149–166, Los Altos, CA, 1986. Morgan Kaufmann.

[33] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[34] J. R. Quinlan. Learning with continuous classes. In *Proc. 5th Australian Joint Conference on AI*, pages 343–348, 1992.

[35] J. R. Quinlan. Combining instance-based and model-based learning. In *Proc. of the 10th International Conference on Machine Learning*, Amherst, MA, 1993.

[36] D. Schuurmans and R. Greiner. Learning to classify incomplete examples. In *Computational Learning Theory and Natural Learning Systems: Addressing Real World Tasks*. MIT Press, 1995.

[37] J. Shavlik, R. Mooney, and G. Towell. Symbolic and neural learning algorithms - An experimental comparison. *Machine Learning*, 6(2):111–143, 1991.

[38] R. E. Stepp and R. S. Michalski. Conceptual clustering: Inventing goal-oriented classifications of structured objects. In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, pages 471–498. Morgan Kaufmann, 1986.

[39] P. R. Thagard, K. J. Holyoak, G. Nelson, and D. Gotchfeld. Analog retrieval by constraint satisfaction. *Artificial Intelligence*, 46:259–310, 1990.

[40] P. D. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 2:369–409, 1995.

[41] P. E. Utgoff. Shift of bias for inductive concept learning. In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, pages 107–148, Los Altos, CA, 1986. Morgan Kaufmann.