k-Nearest Neighbor Classification on First-Order Logic Descriptions

S. Ferilli M. Biba T.M.A. Basile N. Di Mauro
F. Esposito
Dipartimento di Informatica
Università di Bari
via E. Orabona, 4 - 70125 Bari - Italia
{ferilli, biba, basile, ndm, esposito}@di.uniba.it

Abstract

Classical attribute-value descriptions induce a multi-dimensional geometric space. One way for computing the distance between descriptions in such a space consists in evaluating an Euclidean distance between tuples of coordinates. This is the ground on which a large part of the Machine Learning literature has built its methods and techniques. However, the complexity of some domains require the use of First-Order Logic as a representation language. Unfortunately, when First-Order Logic is considered, descriptions can have different length and multiple instance of predicates, and the problem of indeterminacy arises. This makes computation of the distance between descriptions much less straightfoward, and hence prevents the use of traditional distance-based techniques. This paper proposes the exploitation of a novel framework for computing the similarity between relational descriptions in a classical instance-based learning technique, k-Nearest Neighbor classification. Experimental results on real-world datasets show good performance, comparable to that of state-of-the-art conceptual learning systems, which supports the viability of the proposal.

1 Introduction

Classical attribute-value descriptions of objects gained wide acceptance and success in the Machine Learning community because the attributes can be seen as dimensions in a multi-dimensional geometrical space, and the related values as the corresponding coordinates. Hence, every possible object can be univocally mapped onto, and identified by, a single point of the space, being made up of a pre-defined number of features for each of which a specific value is provided. This representation allowed the development of grouping and discrimination techniques that implement learning as an application based on mathematical and geometrical concepts and properties. The core facility of associating objects to points in a geometrical space relies in the possibility of straightforwardly assessing how much two given objects are similar to, or different from, each other as a simple application of the Euclidean distance between the corresponding coordinate vectors (symbolic features can be easily mapped onto discrete numerical coordinates, as well). This allowed the learning techniques to reach high performance, but finds its limit in the rigidity of the descriptions that must identify a fixed number of features in which capturing all possible situations, and that must include all (and, if possible, only) those that are significant for accomplishing the given task.

However, the complexity of some domains cannot be captured by simple attribute-value descriptions, and requires the possibility of including in the descriptions a variable number of objects and features and the ability to express relations between objects. To deal with such domains, First-Order Logic (*FOL* for short) represents a suitable formalism that can overcome the typical limitations shown by propositional or attribute-value representations. As a consequence and tradeoff for its expressive power, however, there is no more a fixed way for comparing two descriptions, but various portions of one description can be possibly mapped in (often many) different ways onto another description. This problem, known as *indeterminacy*, not only causes a significant computational effort when two descriptions have to be compared to each other, but also excludes a straightforward computation of the distance between them, since FOL does not induce a Euclidean space in which reusing consolidated mathematical and geometrical notions. This explains why much less work has been done in the Machine Learning literature on distance-based methods and techniques for FOL de-

978-0-7695-3503-6/08 $25.00 © 2008 IEEE
DOI 10.1109/ICDMW.2008.50

scriptions than for propositional ones.

This paper aims at contributing in this critical area, proposing the adoption of a novel framework, that supports the comparison between FOL clauses, to perform similarity-based Machine Learning in relational domains. This allows many applications, covering both supervised and unsupervised learning, and ranging from (conceptual) Clustering to Instance-based techniques. In particular, it focuses on the k-Nearest Neighbor (k-NN for short) technique, that strongly relies on the availability and quality of similarity measures for classifying unseen observations according to the closest known prototypes. In addition to yielding a similarity evaluation of entire descriptions, the proposed framework allows to compare also description components, suggesting those that are more similar and hence more likely to correspond to each other and this way tackling indeterminacy. Since this concerns the semantic aspects of the domain, and hence there is no precise (i.e., algorithmic) way for recognizing the correct (sub-)formulæ to be associated, the problem is attacked based on the syntactic structure alone.

In the following sections, after presenting preliminary notions about the formalism and a brief recall of related work, the similarity framework will be introduced, from the parameters and corresponding formula on which the framework is based, through similarity criteria for descriptions sub-components, up to the assessment of similarity between whole clauses. Then, a report of the experiments on k-Nearest Neighbor classification in two real-world domains will be provided, before concluding the paper and outlining future work directions.

2 Preliminaries

Logic Programming [13] is the fragment of FOL that is based on formulæ in the form of clauses. It is an important paradigm in Artificial Intelligence, and many first-order Machine Learning systems infer theories in the form of logic programs. Logic programs (or *theories*) are made up of *Horn clauses*, i.e. logical formulæ of the form $l_0 \vee \neg l_1 \vee \cdots \vee \neg l_n$ which is equivalent to $l_1 \wedge \cdots \wedge l_n \Rightarrow l_0$ usually represented in Prolog style as l_0 :- $l_1, \ldots, l_n$ to be interpreted as "l_0 (called *head* of the clause) is true, provided that l_1 and ... and l_n (called *body* of the clause) are all true". The l_i's are *atoms* (i.e., predicates applied to a number of terms equal to their arity); a *literal* is an atom (called *positive* literal) or its negation (called *negative* literal). Two literals are *linked* if and only if they share at least one of their arguments; a clause is linked if and only if any two of its literals can be connected by a chain of pairwise linked literals in the clause. A clause is *range restricted* if and only if all terms appearing in the head also appear in the body. Datalog is a restriction of Prolog where only variables and constants are allowed as terms (i.e., it is syntactically the function-free version of Prolog) [4].

We will deal with the case of linked Datalog clauses, without loss of generality: indeed, linked sub-parts of non-linked clauses can be dealt with separately (because, having no connection between each other, do not contribute any information that is relevant to describe the head), while the *flattening/unflattening* procedures [14] can translate generic first-order clauses (allowing also functions as terms) into Datalog ones and vice-versa. Moreover, we will assume that examples are represented, according to *direct relevance*, as ground (variable-free) clauses where the argument(s) of the head represent the (n-tuple of) object(s) to be classified, the head predicate their class, and the body represents the set of all and only those known literals in the knowledge base that are significant for describing the head, where a literal is relevant if it is (directly or indirectly) linked to the head. Again this is not limiting, since given a general knowledge base it is possible to collect all and only those facts that fulfill such requirement. Observations to be classified will be described in the same way, but with a dummy predicate in the head.

Given two Datalog (sub-)formulæ C' and C'', a *term association* on them is a set of couples of terms, usually written t'/t'', where $t' \in \mathit{terms}(C')$ and $t'' \in \mathit{terms}(C'')$ ($\mathit{terms}(\cdot)$ denotes the set of terms appearing in $\cdot$). In the following, we will call *compatible* two FOL (sub-)formulæ that can be mapped onto each other without yielding inconsistent term associations (i.e., a term in one formula cannot be mapped onto different terms in the other formula).

3 Related Work

Previous work on k-NN in FOL includes *RIBL* [6], a k-NN classifier based on a modified version of the similarity function proposed in [3] for the system *KGB*. The basic idea of the measure used in *RIBL* is that objects are described by values (e.g., size and position) and by their relations to other objects. The similarity between two objects is computed by considering the immediate neighborhood of the objects. For instance, the similarity between two identifiers is determined by the similarity between the sets of facts where these identifiers occur. Also, the similarity between two facts is determined by the similarity between their arguments. If some of these arguments are in turn identifiers themselves, one can get a loop: therefore, a depth parameter is introduced and similarity is only computed up to this depth. Thus, the similarity among objects depends on the similarity of their attributes' values (e.g., the similarity of their size) and the similarity of the objects related to them. The similarity of the related objects in turn depends on the attribute values of these objects and their relation to other objects and so on. Such a propagation poses the problem of indeterminacy in associations, that our technique avoids thanks to the different structural approach. Although the indeterminacy problem could be handled by using some CSP technique (e.g., [16]), our proposal is exploiting the information provided by the peculiar structure of clauses.

RISE [5] is another system that combines a rule classifier with a k-NN technique: when the instance to be classified is not covered by any rule, the distance of the instance to the different rules is evaluated and the instance is classified according to the majority vote of its "neighbour" rules. However, it works in a propositional setting and thus is not directly comparable to a full FOL approach.

More recently, k-RNN, a k-Nearest Neighbour algorithm that works in a FOL setting, has been presented in [11]: in this case the similarity measure between two examples is computed as the ratio of the number of common *saturated clauses* that can be generated by a Mode-Directed Inverse Entailment (MDIE) approach from the examples, over the number of such clauses that can be generated by the new example alone. This formulation makes the measure non-symmetric, due to the denominator involving the training example only, but this odd feature and its consequences are not commented in that work. A particular care is put in parameter setting and fine-tuning to optimize the system efficiency: our technique is not optimized at all, but this will be an issue to be faced in future work. Moreover, our technique does not require additional background information, such as mode declarations in MDIE.

Some work has also been carried out on coupling k-NN with queries on relational databases (e.g., [1]) and on the association of these k-NN with clustering, but it is beyond the scope of this paper, that specifically focuses on the Inductive Logic Programming perspective on relational learning and on pure k-NN.

4 The Similarity Framework

The similarity framework for Horn Clauses on which the work in this paper is based includes the definition of a new similarity function, but its main novelty relies in providing a set of criteria that focus on particular clause components to tackle the problem of indeterminacy while still preserving a considerable amount of information about the description structures. It is syntax-based, and hence totally general, since it does not assume the availability of deep domain-related knowledge for assessing the similarity degree between two descriptions. Here, we briefly recall it from [10].

Like in classical and state-of-the-art distance measures in the current literature, mostly developed in the propositional setting (e.g., those by Tverski, Dice or Jaccard), the evaluation of similarity between two items i' and i'' is based both on the number of common and different features between them [12]:
n, number of features owned by i' but not by i'';
l, number of features owned both by i' and by i'';
m, number of features owned by i'' but not by i'.
However, the new formula does not show the undesirable behaviour of those measures in cases in which n, l or m are 0; expressed in terms of the items to be compared or, equivalently, in terms of the corresponding parameters, it is the following:

$$\mathrm{sf}(i', i'') = \mathrm{sf}(n, l, m) = \frac{l+1}{2}\left(\frac{1}{l+n+2} + \frac{1}{l+m+2}\right) \quad (1)$$

It takes values ranging in the classical spectrum $]0, 1[$, which can be interpreted as the level of likelihood/confidence that the two items under comparison are actually similar. A complete overlapping of the two ($n = m = 0$) tends to the limit of 1 as long as the number of common features grows, whereas in case of no overlapping ($l = 0$) the function will tend to 0 as long as the number of non-shared features grows. The left-hand-side ratio in parentheses refers to item i', while the right-hand-side ratio refers to item i'', which allows to weight them differently if needed (e.g., when comparing a model to an observation). Using equal weight (1/2), as in (1), the function is symmetric with respect to the two items to be compared.

The framework proposed in this work for Instance-based learning exploits repeatedly and pervasively the above formula in various combinations that assign a similarity degree to progressively complex clause components, from terms to atoms to sequences of atoms to whole clauses. The similarity of each component type is based on the similarity of simpler components (only), so that no recursion nor indeterminacy can be present. Empirically, one can note that no single component type is by itself neatly discriminant, but their cooperation succeeds in assigning sensible and useful similarity values to the various kinds of components, and in distributing on each kind of component a suitable portion of the overall similarity, so that the difference becomes ever clearer as long as they are composed one ontop the previous ones. This makes the proposed approach robust to lacks of information due to some of the components.

In FOL formulæ, terms represent specific objects, that are related to each other by means of predicates. Hence, two main levels of similarity can be defined for pairs of first-order descriptions: the *object* level, concerning similarities between the objects referred to in the descriptions (and represented by terms), and the *structure* one, referring to how the nets of relationships in the descriptions overlap (expressed by n-ary predicates applied to terms).

4.1 Object-level Similarity

Consider two clauses C' and C''. Call $A' = \{a'_1, \ldots, a'_n\}$ the set of terms in C', and $A'' = \{a''_1, \ldots, a''_m\}$ the set of terms in C''. When comparing a pair of objects $(a', a'') \in A' \times A''$, a first kind of features to be compared is the set of properties they own (*characteristic features*), usually expressed by unary predicates. For instance, characteristic features for a term representing a person could be `young(X)` or `male(X)`. A corresponding similarity value, called *characteristic similarity*, is obtained by applying (1) to the sets P' of characteristic features related to a' and P'' of characteristic features related to a'':

$$\mathrm{sf}_c(a', a'') = \mathrm{sf}(|P' \setminus P''|, |P' \cap P''|, |P'' \setminus P'|)$$

Also the ways in which terms relate to each other, generally expressed by the position the term holds among the n-ary predicate arguments, determine additional features useful for comparison (*relational features*): indeed, different positions actually refer to different roles played by the objects. For instance, relational features in the predicate `parent(X,Y)` are represented by the roles of the parent (first argument position) and of the child (second argument position). Thus, another similarity value, called *relational similarity*, is based on how many times the two objects play the same or different roles in the n-ary predicates, by applying (1) to the *multi*sets R' of roles played by a' and R'' of roles played by a'' (they are multisets because a term can play the same role in different instances of the same predicate, e.g. a parent of many children):

$$\mathrm{sf}_r(a', a'') = \mathrm{sf}(|R' \setminus R''|, |R' \cap R''|, |R'' \setminus R'|)$$

These values can be combined, so that the overall *object similarity* between a' and a'' is defined as

$$\mathrm{sf}_o(a', a'') = \mathrm{sf}_c(a', a'') + \mathrm{sf}_r(a', a'') \qquad (2)$$

4.2 Structure-level Similarity

While comparison among terms still belongs (can be reduced) to the propositional (attribute-value) setting, checking the structural similarity of two formulæ involves the way in which terms are related by means of atoms built on n-ary predicates. Hence, it is peculiar to the first-order logic setting, and introduces the problem of indeterminacy in mapping (parts of) a formula into (parts of) another one. This is equivalent to the computation of (sub-)graph homomorphisms, a problem known to be *NP*-hard due to the possibility of mapping a (sub-)graph onto another in many different ways. The proposed framework focuses on linkedness (i.e., the fact that two atoms share at least one of their arguments) as a feature on which basing the structural similarity assessment.

The simplest relational components in a first-order logic formula are atoms. Thus, a first problem is computing the degree of similarity between two atoms l' and l''. In this case, linkedness can be exploited 'in-breadth', considering the concept of *star* of an n-ary atom (the multiset of n-ary predicates corresponding to the atoms linked to it by some common term – indeed, a predicate can appear in multiple instances among these atoms). The *star similarity* between two compatible n-ary atoms l' and l'' having stars S' and S'', respectively, can be computed by applying (1) to the number of common and different elements in each of the two stars. However, also the similarity of the objects involved in the two atoms l' and l'' must be taken into account, and hence the star similarity also considers the object similarity for all pairs of terms included in the association θ that maps l' onto l'':

$$\begin{aligned}\mathrm{sf}_s(l', l'') \quad &= \quad \mathrm{sf}(|S' \setminus S''|, |S' \cap S''|, |S'' \setminus S'|) + \\ &\quad + avg(\{\mathrm{sf}_o(t', t'')\}_{t'/t'' \in \theta}) \qquad (3)\end{aligned}$$

Being able to compare two atoms, the next step is comparing sequences of atoms. In this case linkedness can be exploited 'in-depth', considering chains of atoms where each atom is linked to both the previous and the next ones in the chain, but the previous and the next one do not share any argument at all. Such chains, for a clause C, can be determined as all possible paths starting from the root and reaching *leaf* nodes (those with no outcoming edges) in the graph $G_C = (V, E)$ built as follows:

- $V = \{l_0\} \cup \{l_i | i \in \{1, \ldots, n\}, l_i$ built on k-ary predicate, $k > 1\}$ and
- $E \subseteq \{(a_1, a_2) \in V \times V \mid terms(a_1) \cap terms(a_2) \neq \emptyset\}$ where the edges are chosen in such a way to

obtain a *stratified* graph in which the head is the only node at level 0, and each subsequent level i is made up by nodes not belonging to previous levels $j < i$ and having at least one term in common with nodes in the previous level $i - 1$.

When comparing two clauses, their heads (roots of the graphs) are unique (being the only head atom in the clause).

Given two clauses C' and C'' with associated graphs $G_{C'}$ and $G_{C''}$ respectively, and two paths $p' =< l'_0, l'_1, \ldots, l'_{n'} >$ in $G_{C'}$ and $p'' =< l''_0, l''_1, \ldots, l''_{n''} >$ in $G_{C''}$, the *intersection* between p' and p'' is defined as the pair of longest compatible initial subsequences $(< l'_1, \ldots, l'_k >, < l''_1, \ldots, l''_k >)$ of p' and p'', excluding the head. Their *differences* are then defined as the incompatible trailing parts:

$p' \setminus p'' =< l'_{k+1}, \ldots, l'_{n'} > \qquad |p' \setminus p''| = n' - k$

$p'' \setminus p' =< l''_{k+1}, \ldots, l''_{n''} > \qquad |p'' \setminus p'| = n'' - k$

Hence, the *path similarity* between p' and p'' is computed by applying (1) to the number of atoms in the maximal compatible subsequences and in the trailing parts:

$$\begin{aligned} \mathrm{sf}_p(p', p'') &= \mathrm{sf}(n' - k, k, n'' - k) + \\ &\quad + avg(\{\mathrm{sf}_s(l'_i, l''_i)\}_{i=1,\ldots,k}) \qquad (4) \end{aligned}$$

4.3 Clause Similarity

As a final step, consider the case of two clauses C' and C'', whose heads are built on predicates having the same arity. Their bodies can be interpreted as the observations describing the tuple of terms in the head by means of all known facts related to them. Thus, assessing the similarity among those tuples consists in assessing the similarity between the respective bodies. According to the technique presented so far, we need a way to count the number of common and different atoms in the two bodies. The set of common atoms can be considered as their *least general generalization*, i.e. the most specific model for the given pair of descriptions $C = \{l_1, \ldots, l_k\}$ for which there exist two substitutions θ' and θ'' such that $\forall i = 1, \ldots, k : l_i\theta' = l'_i \in C'$ and $l_i\theta'' = l''_i \in C''$, respectively.

Using the classical θ-subsumption generalization model such a generalization is unique and can be computed according to Plotkin's algorithm, but gives some undesirable side-effects concerning the need of computing its reduced equivalent (and also shows some counter-intuitive aspects). For this reason, most ILP learners require the generalization to be a subset of the clauses to be generalized, in which case the θ_{OI} subsumption generalization model [8], based on the Object Identity assumption, represents a supporting framework with solid theoretical foundations to be exploited. The work in [9] shows that the similarity techniques shown so far are also able to guide the generalization procedure in obtaining quickly an accurate approximation of the least general generalization. Under θ_{OI} subsumption, the least general generalization is not unique, but each minimal generalization is already reduced, and hence all of the atoms that make it up are mapped onto different atoms in the generalized clauses, which reduces computation of the common atoms to counting the length of the generalization, and computation of the different ones to subtracting the length of the generalization from that of either clause. The overall clause similarity is computed according to the amount of overlapping and different atoms and terms, taking into account also the star similarity values for all pairs of atoms associated by the least general generalization (which includes both structural and object similarity):

$$\begin{aligned} \mathrm{fs}(C', C'') &= \mathrm{sf}(|C'| - |C|, |C|, |C''| - |C|) \cdot \\ &\quad \cdot \mathrm{sf}(n_o, l_o, m_o) + \\ &\quad + avg(\{\mathrm{sf}_s(l'_i, l''_i)\}_{i=1,\ldots,k}) \qquad (5) \end{aligned}$$

where
$n_o = |terms(C')| - |terms(C)|$;
$l_o = |terms(C)|$;
$m_o = |terms(C'')| - |terms(C)|$.

The first two components are multiplied in order to have a limited influence on the overall similarity, that must be predominantly determined by the similarity of the single atoms.

As to the computational complexity, if n is the number of literals in the longest clause and m is the number of nodes at each level of the clause graph, in the worst case of the atoms being absolutely equally distributed both *in* and *among* levels we have $m = \sqrt{n}$ and hence an order of $\sqrt{n}^{\sqrt{n}}$. However, in more realistic cases, in which atoms are irregularly distributed in breadth and/or depth, and adjacent levels are not completely connected, the complexity moves towards the two extremes $m = 1 \Rightarrow m^{n/m} = 1^n = 1$ and $m = n \Rightarrow m^{n/m} = n^1 = n$.

4.4 example

Let us show the main concepts concerning the similarity framework in the following toy clause (a real-world one would be too complex):

$$\begin{aligned} C : h(a) \text{ :- } & p(a,b), p(a,c), p(d,a), \\ & r(b,f), o(b,c), q(d,e), t(f,g), \\ & \pi(a), \phi(a), \sigma(a), \tau(a), \sigma(b), \tau(b), \phi(b), \\ & \tau(d), \rho(d), \pi(f), \phi(f), \sigma(f). \end{aligned}$$

The set of properties of a is $\{\pi, \phi, \sigma, \tau\}$, for b it is $\{\sigma, \tau\}$ and for c it is $\{\phi\}$. The multiset of roles of a is $\{p/2.1, p/2.1, p/2.2\}$, for b it is $\{p/2.2, r/2.1, o/2.1\}$ and for c it is $\{p/2.2, o/2.2\}$.

The star of $p(a, b)$ is the multiset $\{p/2, p/2, r/2, o/2\}$, while that of $p(a, c)$ is $\{p/2, p/2, o/2\}$.

As to the graph G_C, the head represents the 0-level of the stratification. Then level 1 of the stratification is obtained by introducing directed edges from $h(X)$ to $p(X, Y)$, $p(X, Z)$ and $p(W, X)$. Now the next level can be built, adding directed edges from atoms in level 1 to the atoms not yet considered that share a variable with them: $r(Y, U)$ – end of an edge starting from $p(X, Y)$ –, $o(Y, Z)$ – end of edges starting from $p(X, Y)$ and $p(X, Z)$ – and $q(W, W)$ – end of an edge starting from $p(W, X)$. The third level of the graph includes the only remaining atom, $s(U, V)$ – having an incoming edge from $r(Y, U)$.

The paths in C (ignoring the head) are:

$$\{< p(a, b), r(b, f), t(f, g) >, < p(a, b), o(b, c) >,$$
$$< p(a, c), o(b, c) >, < p(d, a), q(d, e) >\}.$$

5 Experiments

Some experiments were designed to check whether the proposed framework is actually able to give significant similarity hints when comparing two structures, and hence can represent a good way for supporting k-Nearest Neighbor classification on FOL descriptions. All of them were run under WindowsXP Professional on a PC endowed with a 2.13 GHz Intel Dual Core Duo processor and 2GB RAM. The k-NN procedure was implemented in SICStus Prolog v. 3.12. 10-fold cross-validation was used to test the learning effectiveness. The 10 folds were created so that the distribution of examples from the various classes was uniform in each fold, and consequently each of the resulting 10 training and test sets contained approximately 90% and 10%, respectively, of examples from each class.

A real-world domain that requires first-order logic descriptions for capturing the complexity of the observations was choosen for performing the experiments aimed at assessing the applicability and performance of the proposed approach. Specifically, it concerns the descriptions (automatically generated from electronic versions of the documents) of scientific papers layout, according to which identifying the papers' type and significant components[1]. It is made up of 353 descriptions of scientific papers first page layout, belonging to 4 different classes: Elsevier journals, Springer-Verlag Lecture Notes series (SVLN), Journal of Machine Learning Research (JMLR) and Machine Learning Journal (MLJ). The complexity of such a dataset is considerable, and concerns several aspects of the dataset: the journals layout styles are quite similar, so that it is not easy to grasp the difference between them; moreover, the 353 documents are described with a total of 67920 atoms, for an average of more than 192 atoms per description (some descriptions are made up of more than 400 atoms); lastly, the description is heavily based on a *part_of* relation that increases indeterminacy.

The k-NN approach performance was compared to that of a concept learning algorithm embedded in the learning system INTHELEX [7]. Specifically, two versions of such a system were exploited: the classical one (I) and a new one (SF) whose generalization operator was modified to take advantage from the proposed similarity framework for identifying the best subdescriptions to be put in correspondence [10]. The performance was evaluated according to both Predictive Accuracy percentage and F-measure (with parameter 1 in order to equally weight Precision and Recall), to ensure that the performance was balanced between positive and negative examples (since each positive example for a class is a negative example for the other classes, the negative examples are three times the positive ones). Classification figures for the concept learning algorithm, averaged on the 10 folds, are reported in Table 1: *Cl* is the number of clauses in the learned theories; *Gen* the number of generalizations carried out; *Spec*$^+$, *Spec*$^-$ and *Exc*$^-$ the number of specializations (by means of positive atoms, negated atoms and exceptions, respectively). The similarity-driven version outperformed the classical one in all considered parameters (time, learning behaviour, accuracy and F-measure); overall, in the 40 runs it saved 1.15 hours, resulting in a 98% average accuracy (+1% with respect to the old version) and 96% average F-measure (+2% with respect to the old version).

For the k-NN approach, k was set, as usually recommended by the literature, to the square root of the number of learning instances, i.e. 17. Notice that the classification was a multi-class one, so (although k is odd) the nearest neighbours are not bound to a binary classification and ties are possible (indeed, 0.5 errors for SVLN in fold 8 means that two classes were the nearest ones, of which one correct). The results of the k-NN classification performance are summarized in Table 2, detailed for each fold. Some interesting considerations can be drawn upon these figures. First of all, the overall accuracy is 94.37%, which means that few documents were associated to the wrong class, and hence the distance technique is very good in identi-

[1]Available at `http://lacam.di.uniba.it:8000/systems/inthelex/index.htm#datasets`

Table 1. Classification results

		Time (sec.)	Cl	Gen	Spec$^+$	Spec$^-$	Exc$^-$	Accuracy	F1-measure
JMLR	SF	588.97	1.9	8.8	1.1	0	0.6	0.98	0.97
	I	1961.66	1.9	9.1	1.3	0.1	1.3	0.98	0.97
Elsevier	SF	52.92	1	6.3	0	0	0	1	1
	I	303.85	2.1	10.1	2.2	0.2	0	0.99	0.97
MLJ	SF	3213,48	4.7	15.7	3.7	2.5	0.8	0.96	0.94
	I	2974.87	5.2	14.5	4.4	3	2.1	0.93	0.91
SVLN	SF	399	2.6	8.1	0.7	0.4	1	0.98	0.94
	I	662.89	3.3	9.4	2.8	0.9	0.6	0.97	0.93

fying the proper similarity features within the given descriptions. Actually, very often in the correct cases not just the majority, but almost all of the nearest neighbors to the description to be classified were from the same class. Then, note that classes Elsevier and MLJ always have 100% accuracy, which means that the corresponding examples are quite significant and sharply distinguishable. Errors were concentrated in the SVLN and JMLR classes, where, however, high accuracy rates were reached. In detail, of the 13 JMLR wrongly classified observations, 8 were confused with Elsevier ones, and 5 as SVLN ones. Of the SVLN errors, 6 concerned Elsevier, 1 JMLR and 1 was a tie between the correct class and JMLR. This reveals that MLJ is quite distinct from the other classes, while Elsevier, although well-recognizable in itself, is somehow in between JMLR and SVLN, which are also close to each other. Interestingly, mismatchings concerning Elsevier are unidirectional: JMLR and SVLN are sometimes confused with Elsevier, but the opposite never happened; on the other hand, in the case of JMLR and SVLN it is bidirectional, suggesting a higher resemblance between the two.

As to the overall comparison to the concept-learning algorithm, it is very interesting to note that, while high performance on Elsevier and low performance on SVLN are confirmed from the conceptual learning case, for MLJ and JMLR the behaviour of the two approaches is opposite, suggesting somehow complementary advantages of each. Elsevier confirmed full accuracy with respect to the similarity-guided version (the non-guided version had a slightly worse performance), while MLJ improved the performance up to full accuracy in the k-NN approach, gaining 4% over the guided version and 7% over the non-guided one. Conversely, accuracy on the remaining classes neatly decreased of about -8%. This can be explained with the fact that printed journals impose a stricter fulfillment of layout style and standards, and hence their instance are more similar to each other. Thus, the concept learning algorithm is more suitable to learn definitions that generalize on peculiar features of such classes.

Runtime refers almost completely to the computation of the similarity between all couples of observations: it takes an average of about 2sec computing each single similarity, which is a very reasonable time considering the descriptions complexity and the fact that the prototype has not been optimized in this preliminary version.

In order to check whether the novel similarity function was actually useful, or just the procedure for assessing the components similarity determined the performance, a comparison to other measures in the literature was made according to average precision, recall and accuracy for each dataset size, plus some information about runtime and number of description comparisons to be carried out. Results revealed an improvement with respect to both Jaccard's, Tverski's and Dice's measures up to +5,48% for precision, up to + 8,05% for recall and up to + 2,83% for accuracy, confirming that also the basic function improved the state-of-the-art.

We wanted also to compare the performance of our methodology to that of other systems in the literature exploiting k-NN. RISE could not be compared to our methodology, since it works in an attribute-value (propositional) setting, and hence cannot handle descriptions having variable length, different number of occurrences of the same predicates and networked links among description components. As regards the other systems, the comparison was carried out on mutagenesis, a classical ILP dataset [15] in which 188 molecule descriptions must be distinguished into 'active' and 'non-active' ones with respect to mutagenicity. These molecules have been selected by domain experts since classical regression-based techniques are not able to learn useful theories for mutagenicity on them. The dataset consists of 188 molecules, described with a total of 25917 atoms, for an average of nearly 138 atoms per description. In particular, we exploited a version of the dataset where the numeric descriptors were previously discretized into symbolic ones, corresponding to inter-

Table 2. Results of the proposed approach

Fold	1	2	3	4	5	6	7	8	9	10	
JMLR	100	85.71	85.71	84.62	76.92	100	92.31	100	83.33	91.67	Avg: 90.03
errors	0/14	2/14	2/14	2/13	3/13	0/13	1/13	0/13	2/12	1/12	Tot: 13/131
Elsevier	100	100	100	100	100	100	100	100	100	100	Avg: 100
errors	0/5	0/6	0/6	0/5	0/5	0/5	0/5	0/5	0/5	0/5	Tot: 0/52
MLJ	100	100	100	100	100	100	100	100	100	100	Avg: 100
errors	0/11	0/11	0/10	0/9	0/9	0/9	0/9	0/10	0/10	0/12	Tot: 0/100
SVLN	83.33	100	100	100	100	62.5	87.5	92.86	87.5	83.33	Avg: 89.70
errors	1/6	0/5	0/6	0/8	0/8	3/8	1/8	0.5/7	1/8	1/6	Tot: 7.5/70
Overall	97.22	96.43	94.29	94.29	91.43	91.43	94.29	98.57	91.43	94.29	Avg: 94.37
errors	1/36	2/36	2/36	2/35	3/35	3/35	2/35	0.5/35	3/35	2/35	Tot: 20.5/353

vals of the original ranges, by an automatic procedure [2]. On such a dataset, we ran a 10-fold cross-validation experiment, using $k = 13$ (square root of 188), and our technique reached an average Predictive Accuracy of 87.22%, which is far beyond the typical performance of classical conceptual ILP learners (around 70-80%).

As regards RIBL, in the original paper [6] the Authors reported an experiment in which RIBL was provided with progressively larger portions of the dataset, up to the whole dataset, on which the best predictive accuracy was reached: just above 70%, both with and without feature weights (compared to FOIL 6.2, that reached about 62%). In a more recent paper [17], RIBL was endowed with different kernels, and compared to other systems. In the case of RIBL endowed with kNN the performance was 77%, while the best performance of k-NN in that comparison was around 84% (reached by SMD - Sum of Minimum Distances algorithm).

As to the k-RNN system, a direct comparison is not possible because in [11] an extended dataset made up of 205 molecules is exploited. In any case, the Authors report a predictive accuracy of 89.31%, but it is actually only the best accuracy among various experimental results obtained varying the k parameter (between 1 and 20) and the length l of the saturated clause (between 2 and 5). In the 32 (k, l) combinations, only 3 cases slightly outperform our outcome (89.31% for $l = 4$ and $k = 3$ or 4, and 88.25% for $l = 3$ and $k = 2$). In these cases k is always very low with respect to the classical square-root setting, that should be around $k = 15$, where the best performance reached by k-RNN is 85.72%.

6 Conclusions and Future Work

The presence of relations in First-Order Logic causes the problem of indeterminacy in mapping portions of a description onto another one. Hence, the space induced by the descriptions is not Euclidean, and no straightforward way is available for assessing the similarity between two descriptions according to a distance measure. However, some real-world problems require the power of relations to be properly represented, and many approaches in Artificial Intelligence in general, and in Machine Learning in particular, are based on the evaluation of similarity/distance between instances. This paper tackles the case of k-Nearest Neighbor classification, and proposes a novel similarity framework for FOL descriptions based only on their syntactic structure.

Based on the experimental outcomes of such a framework on the real-world task of document classification, it seems a viable solution that does not require deep knowledge of the domain and of the specific descriptors, still providing high performance on a difficult domain, comparable to those of a concept learning algorithm. Also with respect to other systems in the literature based on k-NN classification it shows a better or comparable performance according to Predictive Accuracy.

Future work will concern fine-tuning of the similarity computation methodology, and its application to other problems, such as flexible matching. Further experiments for k-Nearest Neighbour classification on other real-world datasets are currently undergoing, and other scheduled work includes coupling k-NN with clustering: clustering with this measure has already shown very good performance, comparable to that of supervised learning [10], and k-NN could be exploited to classify new instances into one or more of the induced clusters according to their full set of instances or just their prototypes. This would be particularly interesting in dynamic environments such as Digital Libraries management, where documents of unknown class in the repository must be grouped into significant classes and then new incoming documents must be associated to the best groups among those available in the repository.

References

[1] A.W. Ayanso. *Efficient processing of k-nearest neighbour queries over relational databases: A cost-based optimization.* EDT Collection for University of Connecticut, 2005.

[2] Marenglen Biba, Floriana Esposito, Stefano Ferilli, Nicola Di Mauro, and Teresa Maria Altomare Basile. Unsupervised discretization using kernel density estimation. In Manuela M. Veloso, editor, *IJCAI*, pages 696–701, 2007.

[3] G. Bisson. Learning in FOL with a similarity measure. In W.R. Swartout, editor, *Proc. of AAAI-92*, pages 82–87, 1992.

[4] S. Ceri, G. Gottlob, and L. Tanca. *Logic Programming and Databases.* Springer, 1990.

[5] P. Domingos. Rule induction and instance-based learning: a unified approach. In *Proc. of IJCAI-95*, pages 1226–1232. Morgan Kaufmann, 1995.

[6] W. Emde and D. Wettschereck. Relational instance based learning. In L. Saitta, editor, *Proc. of ICML-96*, pages 122–130, 1996.

[7] F. Esposito, S. Ferilli, N. Fanizzi, T. Basile, and N. Di Mauro. Incremental multistrategy learning for document processing. *Applied Artificial Intelligence Journal*, 17(8/9):859–883, 2003.

[8] Floriana Esposito, Nicola Fanizzi, Stefano Ferilli, and Giovanni Semeraro. A generalization model based on oi-implication for ideal theory refinement. *Fundam. Inform.*, 47(1-2):15–33, 2001.

[9] S. Ferilli, T.M.A. Basile, N. Di Mauro, M. Biba, and F. Esposito. Similarity-guided clause generalization. In R. Basili and M.T. Pazienza, editors, *AI*IA-2007: Artificial Intelligence and Human-Oriented Computing*, volume 4733 of *Lecture Notes in Artificial Intelligence*, pages 278–289. Springer, Berlin, 2007.

[10] S. Ferilli, T.M.A. Basile, N. Di Mauro, M. Biba, and F. Esposito. Generalization-based similarity for conceptual clustering. In Z.W. Ras, S. Tsumoto, and D. Zighed, editors, *MCD-2007*, volume 4944 of *Lecture Notes in Artificial Intelligence*, pages 13–26. Springer, Berlin, 2008.

[11] N.A. Fonseca, V. Santos Costa, R. Rocha, and R. Camacho. k-rnn: k-relational nearest neighbour algorithm. In *Proceedings of SAC '08: 2008 ACM Symposium on Applied Computing*, pages 944–948, New York, NY, USA, 2008. ACM.

[12] Dekang Lin. An information-theoretic definition of similarity. In *Proc. 15th International Conf. on Machine Learning*, pages 296–304. Morgan Kaufmann, San Francisco, CA, 1998.

[13] J. W. Lloyd. *Foundations of logic programming; (2nd extended ed.).* Springer-Verlag New York, Inc., New York, NY, USA, 1987.

[14] C. Rouveirol. Extensions of inversion of resolution applied to theory completion. In *Inductive Logic Programming*, pages 64–90. Academic Press, 1992.

[15] A. Srinivasan, S. Muggleton, R. King, and M. Sternberg. Mutagenesis: ILP experiments in a non-determinate biological domain, 1994.

[16] S. Wieczorek, G. Bisson, and M. B. Gordon. Guiding the search in the no region of the phase transition problem with a partial subsumption test. In *Machine Learning: ECML 2006*, volume 4212 of *LNCS*, pages 817–824. Springer, 2006.

[17] Adam Woznica, Alexandros Kalousis, and Melanie Hilario. Distances and (indefinite) kernels for sets of objects. volume 0, pages 1151–1156, Los Alamitos, CA, USA, 2006. IEEE Computer Society.