A Dynamic Approach to Dimensionality Reduction in Relational Learning

Erick Alphonse and Stan Matwin

LRI - Bt 490
Universit Paris-Sud
91405 ORSAY CEDEX
{alphonse,stan}@lri.fr

Abstract. This paper argues that in order to perform data mining on large relational databases with multiple tables, one needs to go beyond the traditional attribute-value learning (AVL) techniques. Inductive Logic Programming lifts the AVL expressivity to the level of first-order logic, well-suited for this task. Several subsets of FOL with different expressive power have been proposed in ILP. The Datalog language is expressive enough to represent realistic learning problems when data is given directly in a multi-relational database. The difficulty lies in the fact that the more expressive power, the more critical the dimensionality of the learning task. The long-term goal of this research is to develop tools that will scale up the ILP learners to make them usable on realistic data mining tasks presented by the KDD community. One of the ways to achieve this goal is to develop effective and efficient feature selection techniques for ILP, hereby decreasing the dimensionality of the problems. We propose the first paradigm that brings feature subset selection to the level of ILP, in languages at least as expressive as (non-recursive) Datalog. The method acts as a filter, which preprocesses the relational data, prior to the model building. An implementation is proposed and successfully applied to the biochemical mutagenesis domain.

1 Introduction

Looking back at the many successes of KDD and Data Mining in the last decade, some researchers ask where should the field go next. Many point to limitations of the representations of the data and the derived knowledge in the existing methods. Most of the existing work uses the attribute-value language (AVL), i.e. each data item describes the same single entity. At the logical level, this AVL representation is equivalent to propositional logic. There are compelling reasons that require the researchers to look beyond this representation. An important one for the use of relational representations is that KDD is best understood invoking the database context. In KDD, more than in Machine Learning, it is natural to perform inductive discovery working on data derived directly from relational databases. In this context, the presence of foreign keys requires the use of a relatively expressive representation, such as Datalog [8].

In Machine Learning, the idea of inducing general knowledge from examples in First Order Logic (FOL) has been known as Inductive Logic Programming for the last 10 years. Many achievements have been accomplished, but researchers have now realized that there exists a dichotomy between expressiveness and efficiency [15]. One of the main difficulties that prevent ILP from tackling large-size problems typical of KDD applications is the dimensionality of the hypothesis space, caused by the fact that search spaces in ILP are considerably larger than in AVL. This implies that the search in ILP learners is much more costly than in AVL learners. Even more importantly, the cost of the coverage test (or conjunctive query problem in relational database terminology) involves logical matching of FOL formulae representing the hypotheses against the training examples. Since the cost of this type of matching is NP-complete, coverage testing, in the worst case, is exponentially more expensive than the same operation in AVL induction. Therefore there is an additional interest in limiting the size of hypotheses spaces to decrease the total cost of coverage tests. In AVL learning, the dimensionality problem has been addressed for years by feature selection techniques. Several successful approaches to feature selection (FS) have been proposed [9, 10], and are widely used not only in research but also in industrial practice, as feature selection functions are included in commercial data mining systems like Mineset and SIPINA. It is only natural to ask if feature selection could be applied to ILP as well. Following Fürnkranz [7], we argue that the usual static ILP approach to limiting the hypothesis space through a restricted hypothesis bias needs to be complemented by a dynamic, data-driven approach similar to the successful FS methods in AVL.

However, the idea of performing FS in an ILP setting runs immediately into a problem: what is an attribute in ILP? All the dynamic FS methods rely on the values of a fixed set of attributes to evaluate their relevance. In ILP, however, there is no fixed set of attributes for a given problem: literals change from example to example, and examples have a variable number of literals. In this paper, we argue that changing the representation from FOL to an attribute-value problem, followed by a purposefully modified version of an attribute-value feature-selecting filter, is an approach to do FS on FOL languages. The change of representation, so-called multi-instance propositionalization, begins to be investigated by several researchers [17, 20, 1]. The main idea of the paper is that this change of representation (from FOL to propositional) defines AVL attributes in terms of a specific fixed set of FOL literals, and then, using well-known FS techniques, we will be able to evaluate the relevance of such literals. If we use as the fixed set of literals those of a given example, we will be able to filter its literals. Nevertheless, we are going to see that the reformulated problem obtained by the multi-instance propositionalization has certain particularities which imply some constraints on the design of the filtering algorithm, which, as far as we know, has not been investigated.

It has to be noted that Lavrač et al. [12] have proposed a feature selection framework in ILP, using a constrained language named Deductive Hierarchical DataBase (DHDB). This language does not allow existential variables and therefore all problems described in DHDB can be compiled into propositional logic in quadratic time. This subset of the first order logic being equivalent to the propositional logic, feature selection techniques can be straightforwardly applied, but this language is too limited to handle current ILP benchmark datasets like mutagenesis (sect. 4) or multi-relational databases [11].

In the next section we introduce, after some notations used in this paper, the multi-instance propositionalization. In section 3, after presenting our paradigm to perform FS in ILP, we show the need to extend classical feature selection algorithms and propose an implementation the paradigm. The method is then empirically validated on the mutagenesis problem, a real-word dataset in ILP used as benchmark section 4. We conclude after discussing the experiments with our approach.

2 Multi-instance Propositionalization

We address learning in the non-recursive Datalog language, which is a non recursive Horn clause language without function symbols other than constants. We use the learning by implication paradigm [5]:
Given a set of positive examples E^+ and negative examples E^-, find a hypothesis, h, such that:

- $\forall e^+ \in E^+, h \geq e^+$
- $\forall e^- \in E^-, h \ngeq e^-$

In other words, we need to find a hypothesis which generalizes or subsumes all positive examples and does not subsume a negative example. Even though logical implication is the most natural subsumption relation, it is not decidable, and therefore we rely on weaker relations like θ-subsumption, widely used in ILP. In the following, we will retain this partial order:

Definition 1 (θ-subsumption [13]). *A clause C θ-subsumes a clause D iff there exists a substitution θ s.t. $C\theta \subseteq D$*

The multi-instance propositionalization [17, 20, 1] is a representation shift that reformulates the FOL learning problem into a particular attribute-value one, which aims at providing the same expressive power [1]. This approach, adapted for different learning systems, is based on the following principle: given an FOL formula P, the propositionalization pattern, each FOL example e is described as a set of attribute-value vectors that have been computed from the set of matching substitutions between P and e. In other words, defining a pattern allows to use its literals to define a fixed set of attributes to re-describe all the FOL examples. Each matching substitution is used to assess the values of attributes, which describe some properties of the initial learning problem.
To clarify the difference between the two kinds of attributes, [18] observes that

[1] It has to be noted that this change of representation strongly differs from Kramer's propositionalization [11] which is a feature contruction process of boolean attributes.

a FOL learning problem can be decomposed into two sub-problems, i.e., a *relational* (or structural) one and a *functional* one. A relational learning problem is concerned with discrimination by predicate occurrence(s) and variable links (equality/inequality between variables), while a functional learning problem is concerned with discrimination through the values of variables seen as attributes. This imposes a partition on variables that may occur in clauses of the learning space: some are referred to as *object* variables whose values do not carry any discriminant information, the others are *attribute* variables, which are assumed to be functions of subsets of object variables. As a consequence, propositionalization shifts the original FOL instance space into an attribute-value lattice in which boolean attributes represent the relational part of the propositionalization pattern, and valued attributes its functional part (see [16] for details).
As a first attempt, we restrict ourselves to relational learning, even if the multi-instance propositionalization is general enough to address both learning problems [18]. The relational problem is typically the non-determinate part of the learning problem, and therefore the part where the dimensionality is the most critical. Consequently, in relational learning, the value of each attribute for a particular substitution indicates the matching or not of its corresponding predicate. The new instance space is a boolean instance space only.

2.1 Propositionalization of the Training Set

We briefly describe the propositionalization process through an example as it is similar to the one defined in [1], althought for another purpose. We define the *pattern of propositionalization* P as the variabilization of a given positive example, under the considered partial order.
Each literal of P defines a boolean attribute that will be used to reformulate each FOL example e (positive or negative) of the training set as a set of boolean vectors. A boolean vector is constructed for each substitution σ_k that matches a subset of P on a subset of e. For convenience, we will not distinguish in the remainder of the paper a matching (substitution) from its associated boolean vector.

Example 1. Let us consider the non-recursive Datalog space ordered by θ-subsumption. Let E, and E' be two positive examples and NE be a negative example of the target concept:
$E' : c(a) \leftarrow p(a,b), q(b), q(a), r(c)$
$E : c(a) \leftarrow p(a,b), p(b,c), q(c), q(a)$
$NE : c(a) \leftarrow p(a,b), p(b,c), q(b), q(a)$
To redescribe the whole dataset, we use E' to construct the pattern P. We build $\mathcal{P}$ similarly to [18] as the following variabilization of E':

$$\mathcal{P} : c(U) \leftarrow p(V,W), q(X), q(Y), r(Z).$$

$\mathcal{C}$ is set to the conjunction of equality constraints between pairs of variables of $\mathcal{P}$ which are satisfied in E', i.e., $\mathcal{C} : U = V, U = Y, V = Y, W = X$.

Given P, we build the new instance space showed in Table 1. Let us more closely consider how example E is reformulated. The propositionalization algorithm searches for all substitutions which, when applied to literals of P, will result in literals belonging to E. The propositionalization process first builds the substitution $\sigma_{E,1} = \{U/a, V/a, W/b, X/a, Y/c\}$. Notice that literal $r(Z)$ of $\mathcal{P}$ has not been matched to any literal of E. Only constraint $U = V$ of P is satisfied for this substitution. When searching for another possible matching for literals $q(X)$ and $q(Y)$ with literals of E, one gets another substitution $\sigma_{E,2} = \{U/a, V/a, W/b, X/c, Y/a\}$, with constraints $U = V, U = Y$ and $V = Y$ of P. Another matching of literal $p(U, V)$ with E yields $\sigma_{E,i} = \{U/a, V/b, W/c, X/c, Y/a\}$, with constraints $U = Y, W = X$ of P. In our example, five other substitutions (and therefore five other boolean vectors) are obtained when searching among all possible partial matchings of variables of $\mathcal{P}$ with constants of E is completed.

Table 1. The tabular representation of a FOL problem

P	$c(U)$	$p(V,W)$	$q(X)$	$q(Y)$	$r(Z)$	$U=V$	$U=Y$	$V=Y$	$W=X$
σ_P	1	1	1	1	1	1	1	1	1
$\sigma_{E,1}$	1	1	1	1	0	1	0	0	0
$\sigma_{E,2}$	1	1	1	1	0	1	1	1	0
...									
$\sigma_{E,i}$	1	1	1	1	0	0	0	0	1
...									
$\sigma_{NE,1}$	1	1	1	1	0	1	0	0	1
$\sigma_{NE,2}$	1	1	1	1	0	1	1	1	1
...									
$\sigma_{NE,j}$	1	1	1	1	0	0	0	1	0
...									

2.2 New Learning Task

As each vector represents a matching of P's literals to the ones of the FOL examples, each reformulated example is described by a set of vectors more general than or equal to P. Therefore, by construction, the propositionalization pattern P is represented by the bottommost element of the new instance space (σ_P in table 1). There exists a one-to-one mapping between a vector and a FOL description where the syntactic representative is simply a subset of the chosen propositionalization pattern.

As pointed out in [20], the learning task is no longer to induce a FOL concept consistent with all positive and negative boolean vectors but is now a *multi-instance problem*[6]:

Definition 2. *The reformulated learning task consists in finding a concept that covers for each FOL positive example at least one of its associated boolean vectors (completeness) and none of the boolean vectors associated to any FOL negative example (correctness).*

3 Feature Selection in ILP

As we have shown, the multi-instance propositionalization (MIP[2]) is based on a definition of a pattern to redefine the original relational problem. We use the set of P's literals as set of attributes, and we use each substitution matching literals of P to a given example to re-describe this example in terms of the set of attributes. Using filtering (feature selection) techniques adapted to work with the reformulated problem, one can select features that correspond to a selection of literals of the propositionalization pattern. If we set a positive[3] example as the propositionalization pattern and select a subset of literals as proposed, we will perform a feature subset selection of this positive example.

However, as pointed out by [17, 4], attribute-value algorithms working on the reformulated problem must deal with data of exponential size wrt the FOL problem. For example, under θ-subsumption and given a pattern P, a FOL example is theoretically to be reformulated as a set of ${n_i}^{m_i}$ vectors, where n_i and m_i are the number of occurrences of predicate symbol p_i in the FOL example e and in P, respectively. For instance, in the mutagenesis dataset, described in section 4 the potential number of matchings is 40^{40}. But this set is highly redundant [1] and only few vectors are indeed sufficient represent the whole instance space. Algorithms can be designed to approximate this set of non-redundant vectors in order to cope with the intractability of the MIP. Indeed, this set is known to be the set of the most specific vectors in the boolean lattice-like instance space (the nearest-misses and nearest-hits of the pattern). An approximation can be achieved by working with a subset of vectors whose elements are as close as possible to the non-redundant, minimal elements. We will refer to this approach as *bounded* multi-instance propositionalization.

Therefore, we propose the following overall scheme for selecting features in ILP:

1. set a not-yet-filtered positive FOL example as the propositionalization pattern and perform a bounded MIP of the relational problem
2. apply a propositional feature selection filter, upgraded for handling MIP (the choice of a relevant filter will be discussed below)
3. output the filtered example, by mapping the selected features onto the relevant literals
4. reiterate with each remaining not-yet-filtered positive examples of the relational database

[2] We will indifferently refer as MIP to both the change of representation process and the result of this process (a multi-instance problem, see section 2.2)

[3] In the learning by implication paradigm, it does not make sense to filter negative examples since the more specific they are, the more informative is the training set.

The result of steps 1-4 represents a reduced FOL representation of the original problem. This reduced representation is then given, as in the AVL FS setting, to an FOL learner to obtain the final solution. In sections 4.1 - 4.2, in order to empirically evaluate the paradigm, we are going to propose a simple implementation of the above steps. We will discuss its limitations in section 5.

3.1 Bounded Multi-instance Propositionalization

One of the simplest bounded propositionalization schemes has been proposed by Sebag et al. [18]. They apply a stochastic process where k matchings are selected to yield a bounded reformulated problem, with k being a user-supplied parameter. This approach has been successfully used in the learning system STILL.

For the stochastic selection, we use, similarly to STILL, a uniform sampling with replacement. This algorithm works just like the propositionalization described in Sect. 2. Instead of an exhaustive search of all the matchings for a given literal, one picks one of them randomly. Therefore, given k, the number of vectors to be computed, the time complexity is $\mathcal{O}\left(k\,|P|\,|e|\right)$, with P and e being the propositionalization pattern and the example respectively.

3.2 Feature Selection of the Reformulated Problem

In our approach, feature selection is performed by first reformulating the relational problem, and then by applying feature subset selection techniques. Consequently, the filter methods used for feature selection must perform well in the context of:

- bounded multi-instance propositionalization
- learning from implication

More precisely, these two approaches combined will inevitably produce noise, and therefore, the issue of noise resistance has to be addressed while designing a filter algorithm. First of all, it has been pointed out that the underlying multi-instance representation of the reformulated problem could be seen as a class-noisy representation of the positive data [2, 3]. According to definition 2, each positive FOL example is represented by a set of vectors, such that covering only one of them is sufficient to cover the FOL example. This problem has been studied in the ILP community and relaxing the completeness is typically applied [20, 1].

We want to focus on the particular source of noise that is the bounded MIP, which induces both class and attribute noise. On one hand, the propositionalization shifts the initial instance space into a boolean one bounded from below by the pattern, each reformulation of FOL examples being casted onto this new space. For instance, reformulating a negative FOL example will produce a set of negative boolean vectors. The converse does not necessarily hold while reformulating a positive example. Indeed, in the disjunctive learning case, the pattern

and the positive example may not belong to the same sub-concept. That is, some literals discriminate between the sub-concept of P and the sub-concept of the example. These literals, used as attributes to describe the several reformulations of the example will not be matched by definition. Therefore the MIP will produce irrelevant and over-general [4] vectors which do not have to be taken into account. In this manner, propositionalization introduces class-noise for these particular positive examples which are tagged positive in the training set but which really are negative examples with respect to the pattern.
On the other hand, bounded propositionalization copes with the intractability of the change of representation by approximating the set of the most specific vectors. This approximation will produce a generalization of the most specific vectors. This generalization can be viewed as obtaining the most specific vectors from a source of noise, in which some attributes' values "true" have been flipped to values "false". Due to our particular learning setting, the attribute-noise impact differs between negative and positive examples. For the former, the so generalized negative examples are still negative. For the latter, flipping true to false can transform the positive vector into a negative one.

This last property seems to invalidate any pruning scheme able to cope with both the class-noise and attribute-noise of positive examples. Moreover, all positive examples other than the pattern are potentially noisy. As far as we know, feature subset selection in a context of a noisy MIP has not been investigated yet. Therefore, as a first attempt to bring feature selection techniques to ILP, we are going to present, in the next section, a simple Relief-like algorithm, which will not take into account positive examples other than the pattern, and therefore will deal with the problem of the attribute-noisy negative examples only.

A Relief-based Filter for the Reformulated Problem Relief has been shown to successfully handle attribute-noise [9]. This algorithm is fast and reliable for detecting relevant, correlated attributes, both nominal and ordinal. Relief is inspired by instance-based learning, observing how attributes differ among instances to assess their relevance. It outputs a ranked list of attributes, maintaining a weight for each feature, which is increased if the feature seems relevant and decreased if irrelevant. The effective selection of relevant features is made by setting a threshold, usually set to zero when no prior knowledge is available.

Using Relief in an ILP context has been already discussed in [14] to prune irrelevant branches of refinement graphs. Although we use a Relief-like filter for a completely different task, we share the same weight update procedure. To assess the relevance of a particular attribute, we can just look at the discrepancy between the attribute values of the pattern and the negative vectors. As the former is the bottommost element of the boolean lattice, i.e. all attributes are set

[4] This case is exemplified by the FOL learning problem described in example 1. It can be seen that a conjunctive solution does not exist in the Datalog language. All vectors obtained by propositionalizing the positive FOL example are more general than the negative vector $\sigma_{NE,2}$.

to one, the relevance of an attribute can be assessed from the negative examples only. So, the gain of an attribute wrt a negative vector depends whether the attribute is set to one or to zero. In our learning setting [5], if the attribute is one, it can not be used to discriminate the negative vector, hence its gain is negative (-1). In the same way, its gain is positive (+1) if its value is zero. The filtering algorithm is shown figure 1.

filtering(k,E^-,P)
% returns an empirically relevant selection of P's literals
  Set all weights W[i] to 0.0
  For each FOL negative example e **in** E^- **do**
    Repeat k **times**
      Compute *one* vector v by propositionalizing e wrt P
      For i := 1 **to** all_attributes **do**
        $W[i] := W[i] + gain(v)$
return all attributes whose the weight is positive.

Fig. 1. Filtering Algorithm

All FOL negative instances are used[6]. For convenience, we can consider the weights normalized between [-1;+1]. If we analyze the two extremal weights, we can see that an attribute with value true in every negative example will have a weight equal to -1, as it can not be used at all to solve the discrimination task. An attribute not belonging to any negative vectors will get a weight +1, as it is very discriminating.

The time complexity of this algorithm is linear in k, in the number of negative examples and in the complexity of computing one vector by propositionalization. In the case where the sampling scheme described in Sect. 3.1 is used, the total complexity is $\mathcal{O}\left(k\,|E^-|\,|P|\,|e|\right)$. Due to the fact we do not use the positive examples, we require neither space in k, nor in the size of the training set, that is to say, the space complexity is constant. This feature will be discussed in section 5.

The filter at work In order to illustrate the filtering process, we are going to filter the positive FOL example E', given as the propositionalization pattern in the example 1. According to section 3.2, for each negative vector computed, each attribute whose the value is true will have a negative gain. Considering only the vectors given in example 1 (as a result of bounded MIP), we have the results given in table 2, the last row showing the normalized weight of each attribute. The effective selection of attributes is performed by selecting those whose the weight is positive, i.e. : $r(Z)$ and $U = Y$.

[5] When learning by implication, only literals from the bottommost clause's body are used for the discrimination task.

[6] In the original version of Relief, only a fixed number of instances picked at random is used. Although this results in a high variance, it is an interesting approach when dealing with a large number of instances

Therefore, given the relational problem and the following example as the pattern:

$$E' : c(a) \leftarrow p(a, b), q(b), q(a), r(c).$$

The current implementation will produce the following filtered example[7]:

$$E' : c(a) \leftarrow q(a), r(c).$$

Table 2. The output of the filter algorithm

P	$c(U)$	$p(V,W)$	$q(X)$	$q(Y)$	$r(Z)$	$U = V$	$U = Y$	$V = Y$	$W = X$
$\sigma_{NE,1}$	-1	-1	-1	-1	+1	-1	+1	+1	-1
$\sigma_{NE,2}$	-1	-1	-1	-1	+1	-1	-1	-1	-1
$\sigma_{NE,j}$	-1	-1	-1	-1	+1	+1	+1	-1	+1
	-1	-1	-1	-1	+1	-0.33	+0.33	-0.33	-0.33

4 Experiments

To evaluate the approach we use the ILP system Propal [1] as learning algorithm.

We have evaluated the approach by performing experiments on the two Mutagenesis datasets, well-known ILP problems used as benchmark tests. In these problems, each example consists of a structural description of a molecule as a definite clause. The molecules have to be classified into mutagenic and non-mutagenic ones. The representation language used has been defined from background knowledge B_0, which uses only relational literals, tackling nominal and ordinal arguments as constants (see [19] for a detailed explanation). In a few words, positive and negative examples of the target concept are molecules described in terms of atoms (between 25 and 40 atoms) and bonds between some of these atoms.

One of the two datasets is "regression-friendly", in which a good regression analysis can be performed, composed of 188 molecules, and the other one, "regression-unfriendly", composed of 42 molecules. The experimental protocol is the one provided in [19]. The accuracy of the learned theory for the regression-friendly dataset (RF) is evaluated by a 10-cross-validation (the 10 folds being already given), and the accuracy on the regression-unfriendly (RU) is evaluated by a leave-one-out procedure. Each learning time is calculated by performing learning on the whole dataset.

Table 3 compares the Propal's performance on the Mutagenesis datasets filtered and not filtered. The time to filter the two databases is negligible. Due

[7] This representation of the final example depends of the language of representation used.

Table 3. Comparison of the PROPAL's performance on the original datasets and the filtered ones

	RF		RU	
	Accuracy (%)	Time (s.)	Accuracy (%)	Time (s.)
Filtered ($k = 100$)	85.54 ±1.3	69 ±26	80.71 ±1.75	6.34 ±1.61
Filtered ($k = 500$)	85.50 ±1.73	110 ±1	82.61 ±3.73	5.96 ±0.85
Not filtered	81.80	290	71.4	10

to the stochastic process of the bounded propositionalization used, each accuracy and time have been averaged over 10 runs; the standard deviation is given.
As expected, we can see that the performance of PROPAL has been improved, both in accuracy and time. It is interesting to notice the small standard deviation obtained for the accuracy, although the total number of vectors extracted is really small compared to the size of the matching space in mutagenesis. That could be an evidence that this space is highly redundant, and the filtering step does not suffer from the poor bounded propositionalization scheme, at least for the mutagenesis problems. The deterioration of the performance for the case with $k = 500$ is most likely due to the injection of noise into the data by working with a larger sample of the noisy instance space.

5 Discussion

Several remarks are in order to summarize the implementation of the paradigm used for the validation. Firstly, we use a very simple bounded MIP scheme which does not depend on the application and the structure of the space of matchings being searched. There is a need for an informed sampling scheme here, one which would take into account the partial ordering between instances in order to extract vectors as specific as possible and possibly particularities of the application. Secondly, we can observe that the parameter k, specifying the size of the example space sampled from the entire search space, is arbitrary in the approach described here. At this point, we do not have a good grasp of the range of values of k which will allow efficient and yet precise learning from the sample. We can observe that although one could anticipate k to be large, our experiments indicate that even a very small k (wrt the total size of the matching search space) is adequate, at least in the mutagenesis application. Thirdly, with the upgrade of feature selection techniques, there is also an upgrade of the filter algorithm, which has to take into account our particular settings. Tackling a bounded MIP needs to be further investigated. We proposed here a Relief-like algorithm for its ability to cope with attribute-noise and numerical attributes (not yet evaluated). The method described does not use any positive examples other that P. Clearly, there is room for improvement here: use of positive examples should allow for a tighter, more focused search space, and consequently a better approximation of the relevant literals. However, it is worth considering it further due to its low complexity, and its constant space requirement which could one allow to handle arbitrarily large sample size (here, k in sect. 3.1).

6 Conclusion

FOL problems typically have to cope with severer dimensionality problems than those in AVL learning. In this paper, we have proposed a general paradigm to allow feature subset selection techniques to be applied in ILP, in languages at least as expressive as Datalog, in order to cope with them.
Our design criteria focused on the ability to handle FOL problems expressive enough to support KDD applications. This ability means that a FOL representation must include free, existentially quantified variable. As far as we know, it is the first feature selection method which works with reasonably expressive subsets of FOL. Moreover, it is applicable theoretically to any partial ordering.

The main idea is to apply a change of representation transformation from FOL to a propositional representation, then to perform feature selection on this representation, and to map the reduced feature set back to the original FOL features (literals), thereby resulting in a reduced FOL representation. In this paradigm, the feature selection techniques work as a front end filter, applied prior to the model building. The reformulated problem which has to be filtered is a noisy MIP, which has not been investigated yet by the machine learning community. Further research is needed to bring effective and efficient techniques to ILP. One of the possibilities is to develop a model of noise introduced by the change of representation as described in Sect. 3.2, and then to come up with an AVL FS under this model, maybe following [3].

In order to evaluate the relevance of the approach, we have proposed a simple implementation, and we have performed experiments on two datasets of mutagenesis, a real-word biochemical domain, considered as a benchmark test in ILP. The results are encouraging showing an improvement both in time and accuracy.
We plan to further investigate the validity of the approach on problems presented by the KDD community still considered out of the scope of current ILP techniques.

Akcnowledgements

We are very grateful to Michèle Sebag and Céline Rouveirol for discussing their works with us. The support of the Natural Sciences and Engineering Research Council of Canada and of the Computing and Information Technologies Ontario are kindly acknowledged.

References

1. E. Alphonse and C. Rouveirol. Lazy propositionalization for relational learning. In W. Horn, editor, *Proc. of the 14th European Conference on Artificial Intelligence (ECAI'2000)*, pages 256–260. IOS Press, 2000.

2. Avrim Blum and Adam Kalai. A note on learning from multiple-instance examples. *Machine Learning*, 30:23–29, 1998.
3. Y. Chevaleyre and J.D. Zucker. Noise-tolerant rule induction for multi-instance data. In *ICML 2000, Workshop on Attribute-Value and Relational Learning*, 2000.
4. L. de Raedt. Attribute-value learning versus inductive logic programming : The missing link. In D. Page, editor, *Proc. of the 8th International Workshop on Inductive Logic Programming*, pages 1–8. Springer Verlag, 1998. Extended abstract.
5. Luc de Raedt. Logical settings for concept-learning. *Artificial Intelligence*, 95(1):187–201, 1997.
6. T. Dietterich, R. Lathrop, and T. Lozano-Perez. Solving the multi-instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89:31–71, 1996.
7. J. Fürnkranz. Dimensionality reduction in ILP: A call to arms. In Luc de Raedt and S. Muggleton, editors, *Proceedings of the IJCAI-97 Workshop on Frontiers of Inductive Logic Programming*, pages 81–86. Nagoya, Japan, August 1997.
8. Jorg-Uwe Kietz, Ulrich Reimer, and Martin Staudt. Mining insurance data at swiss life. In *The VLDB Journal*, pages 562–566, 1997.
9. K. Kira and L. A. Rendell. A practical approach to feature selection. In *Proc. of the Ninth International Conference on Machine Learning.*, pages 249–256. MK, 1992.
10. Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1–2):273–323, 1997.
11. S. Kramer, B. Pfahringer, and C. Helma. Stochastic propositionalization of non-determinate background knowledge. In D. Page, editor, *Proc. of the 8th International Workshop on Inductive Logic Programming*, pages 80–94. Springer Verlag, 1998.
12. Dragan Gamberger Nada Lavrač and Viktor Jovanoski. A study of relevance for learning in deductive databases. *Journal of Logic Programming*, 40(2/3):215–249, 1999.
13. G. Plotkin. A note on inductive generalization. In *Machine Intelligence*, volume 5. Edinburgh University Press, 1970.
14. U. Pompe and I. Kononenko. Linear space induction in first order logic with RELIEFF. In R. Kruse, R. Viertl, and G. Della Ricci, editors, *Mathematical and Statistical Methods in Artificial Intelligence, CISM Course and Lecture Notes 363*, pages 185–220. Springer-Verlag, 1995.
15. C. Rouveirol. Expressivité/efficacité : le dilemne de la programmation logique inductive. LRI, décembre 2000. HDR.
16. M. Sebag. Resource bounded induction and deduction in fol. In *Multi Strategy Learning*, 1998.
17. M. Sebag and C. Rouveirol. Induction of maximally general clauses consistent with integrity constraints. In S. Wrobel, editor, *Proceedings of the 4th International Workshop on Inductive Logic Programming*, volume 237 of *GMD-Studien*, pages 195–216. Gesellschaft für Mathematik und Datenverarbeitung MBH, 1994.
18. M. Sebag and C. Rouveirol. Resource-bounded relational reasoning: Induction and deduction through stochastic matching. *Machine Learning*, 38(1/2):41–62, 2000.
19. A. Srinivasan, S. Muggleton, R.D. King, and M.J.E. Sternberg. Mutagenesis: ILP experiments in a non-determinate biological domain. In S. Wrobel, editor, *Proceedings of the 4th International Workshop on Inductive Logic Programming*, volume 237 of *GMD-Studien*, pages 217–232. Gesellschaft für Mathematik und Datenverarbeitung MBH, 1994.

20. J.-D. Zucker and J.-G. Ganascia. Changes of representation for efficient learning in structural domains. In *Proc. of 13th International Conference on Machine Learning*. Morgan Kaufmann, 1996.