, , 1–18 ()
© Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.

Filtering Multi-instance Problems to Reduce Dimensionality in Relational Learning

ÉRICK ALPHONSE alphonse@lri.fr
LRI - UMR 8623 CNRS
Bât 490 - Universit Paris-Sud
91405 ORSAY CEDEX FRANCE

STAN MATWIN stan@site.uottawa.ca
SITE - University of Ottawa
Ottawa Ontario
K1N 6N5 CANADA

Received November 12, 2002; Revised February 11, 2002

Editor:

Abstract. Attribute-value based representations, standard in today's data mining systems, have a limited expressiveness. Inductive Logic Programming provides an interesting alternative, particularly for learning from structured examples whose parts, each with its own attributes, are related to each other by means of first-order predicates. Several subsets of first-order logic (FOL) with different expressive power have been proposed in Inductive Logic Programming (ILP). The challenge lies in the fact that the more expressive the subset of FOL the learner works with, the more critical the dimensionality of the learning task. The Datalog language is expressive enough to represent realistic learning problems when data is given directly in a relational database, making it a suitable tool for data mining. Consequently, it is important to elaborate techniques that will dynamically decrease the dimensionality of learning tasks expressed in Datalog, just as Feature Subset Selection (FSS) techniques do it in attribute-value learning. The idea of re-using these techniques in ILP runs immediately into a problem as ILP examples have variable size and do not share the same set of literals. We propose here the first paradigm that brings Feature Subset Selection to the level of ILP, in languages at least as expressive as Datalog. The main idea is to first perform a change of representation, which approximates the original relational problem by a multi-instance problem. The representation obtained as the result is suitable for FSS techniques which we adapted from attribute-value learning by taking into account some of the characteristics of the data due to the change of representation. We present the simple FSS proposed for the task, the requisite change of representation, and the entire method combining those two algorithms. The method acts as a filter, preprocessing the relational data, prior to the model building, which outputs relational examples with empirically relevant literals. We discuss experiments in which the method was successfully applied to two real-world domains.

Keywords: Relational Learning, Feature Subset Selection, Propositionalization

1. Introduction

What needs to be done to bring data mining (DM) and knowledge discovery in databases (KDD) into the realm of everyday database practice? What are the constraints of the existing KDD approaches and systems that stand in the way of this goal? Many point to the limitations of the representations of data and derived knowledge in the existing methods. Most of the existing DM/KDD work is in attribute-value learning (AVL), where examples are represented by vectors of fixed length in which each coordinate represents a value of a given attribute, and all the attributes describe the same single entity. At the logical level, this AV representation is equivalent to propositional logic. There are compelling reasons that require researchers to look beyond this representation.

From the database perspective, two issues need to be addressed. Firstly, many domains of application do not lend themselves naturally to the simple AV representation, as problems in these domains have an inherent structure, in which parts, represented by different entities, are combined and constrained by means of rich sets of relationships. This occurs often in biological and chemical applications. Some of the examples of such naturally structural domains are: crystallography (Conklin, 1995), mutagenicity (Srinivasan et al., 1994), carcinogenicity (Lee et al., 1998). Similar needs for structural representations have also been noticed in financial applications - e.g. (Kietz et al., 1997). It is therefore important for the field of KDD to use representations at the right level of expressivity for these important structural domains. First-order logic provides a well understood representation in which it is easy to express structural descriptions.

Secondly, an important goal of KDD is to perform discovery working directly with the data as represented in a relational database. In this context, the presence of foreign keys requires the use of a relatively expressive representation. The issue of the level of this representation has been researched carefully in the literature. An early approach, known as constrained Deductive Hierarchical Data Bases (constrained DHDB, used in (Lavrač and Džeroski, 1994)) is not sufficient for KDD, as free variables are not supported. Later on, Datalog relaxes these constraints by allowing a representation of clauses with free variables and constants (but no structured terms). The work presented here is therefore situated at the level of non-recursive Datalog clauses.

From the Machine Learning perspective, 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 trade-off between expressiveness and efficiency (Rouveirol, 2000). If there is no generally accepted theoretical measure of the dimensionality of ILP problems (Fürnkranz, 1997), researchers agree that the major factor is the complexity of the hypothesis space. Hypothesis spaces in ILP are considerably larger (even infinite under θ-subsumption) than in AVL. The standard solution to the overgrown hypothesis space is the idea of a declarative bias. Traditionally in ILP, declarative (static) biases are defined to constrain the hypothesis language. Such biases were mainly constraints on the hypothesis language, introduced by the user (Nédellec et al., 1996). Their effect may be detrimental for the learning task: either the hypothesis language can become too restrictive to express the target concept, or too general so that their constraining effect becomes insignificant.

In this paper, we focus on reducing another factor influencing dimensionality of ILP: the complexity of the example space. The complexity of the example space is understood here not as the number of examples (as in (Fürnkranz, 1997)), but as the size of the examples in terms of the number of literals by which the examples are expressed. Such example complexity also contributes to the dimensionality of the ILP task for the following reasons. Firstly, the hypothesis language is based on the example language, so the simpler the example language, the smaller the hypothesis space. Secondly, the coverage test in ILP (equivalent to the conjunctive query problem in relational database terminology in languages typically handled in ILP) is NP-complete, and therefore, in the worst case, is exponentially more expensive in the size of the examples than the same operation in AVL. Thirdly, the presence of irrelevant literals in the examples description may mislead the heuristic search by causing plateau phenomena (Silverstein and Pazzani, 1991). All the three situations make it interesting to decrease the size of the examples by eliminating some literals from them.

In AVL learning, the dimensionality problem has been addressed for years by feature selection techniques. Several successful approaches to feature selection (FSS) have been proposed (Kira and Rendell, 1992, Kohavi and John, 1997, Liu and Motoda, 1998), 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, SAS and SPSS. It is only natural to ask if feature selection could be applied to ILP as well. Following Fürnkranz (1997), 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 FSS methods in AVL.

However, the idea of performing FSS in an ILP setting runs immediately into a problem: what is an attribute in ILP? All the dynamic FSS methods rely on the values of a fixed set of attributes, shared by all examples, 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.

As far as we know, the only work related to filtering irrelevant literals out of relational examples has been proposed by Cohen (1995), in the context of text mining. Cohen proposes to filter out literals (here words or relations between words) that have a low frequency in the learning corpus. This technique is typically used in text mining and it makes sense solely in this context.
Our approach to FSS in ILP is more general: it works with any relational examples that can be expressed as non-recursive Datalog clauses. Seen as a black box, our FSS filter processes FOL examples one by one, and removes from them literals which are judged irrelevant. This is achieved by first approximating the original FOL problem as an attribute-value problem, and then applying a purposefully modified version of an attribute-value feature-selecting filter. The new, reduced propositional problem will be converted back to the relational representation, resulting in a empirically equivalent dataset with reduced dimensionality.
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 FSS 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. The change of representation, so-called multi-instance propositionalization, begins to be investigated by several researchers (Zucker and Ganascia, 1996, Sebag and Rouveirol, 1996, Alphonse and Rouveirol, 2000). We are going to show that it has certain properties which result in interesting constraints on the design of the FSS algorithm. Feature selection algorithms working under these constraints, as far as we know, have not been investigated in the literature.

In the next section we describe, using generic components, the main idea of our approach. We then introduce, after some notations used in this paper, the change of representation (multi-instance propositionalization). In section 4, after presenting our paradigm to perform FSS in ILP, we show the need to extend classical feature selection algorithms and propose an implementation of the paradigm. The method is then empirically validated on two real-life problems: the mutagenesis problem, a real-word dataset in ILP used as benchmark (section 6), and another real-life problem in the area of document understanding (Esposito et al., 1994). We conclude after discussing the experiments with our approach, and presenting several specific, interesting future research problems stemming from the work presented here.

2. Overview of the Architecture

The general idea of our approach is illustrated in Fig. 1. First of all, both the inputs and the outputs of our method are relational examples, which we view as sets of literals. In that sense, the method is similar to AVL-type feature selection, where the inputs and the outputs use the same representation language, and the learning system works on the output of the FSS process. All AVL feature selection methods rely on a fixed set of attributes, the same for all examples, to evaluate their relevance for the learning task. In order to bridge the gap between the flexible format of ILP examples and the fixed representation requirement of FSS in AVL, the idea is to filter the literals of *one FOL example at a time*. As shown in Fig. 1, we use the set of literals of the example being filtered as a fixed set of attributes, and re-describe the whole relational problem by means of these attributes.

Our change of representation converts each other relational example into a set of AV examples. This makes us view the propositional representation as a multi-instance problem: a bag of AV examples is obtained from a single relational example, and it inherits the label of its relational source. This so-called multi-instance propositionalization (MIP) has been investigated by several researchers (Zucker and Ganascia, 1996, Alphonse and Rouveirol, 2000). In MIP, each ILP example is transformed into a set of fixed-length vectors of AVL attributes. The vector format has a boolean attribute for each literal of the example being filtered. Each boolean attribute describes the fact that its corresponding literal matches (under some substitution) a literal of the transformed ILP example.

Having obtained a fixed format representation of the initial relational problem from the point of view of an ILP example, we are able to evaluate the relevance of

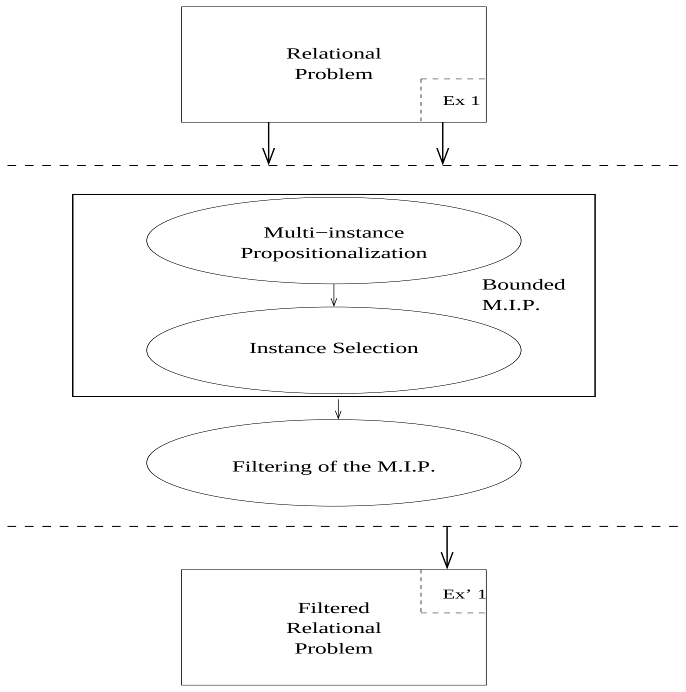

Figure 1. Filtering an relational example

each attribute in the AVL setting, and consequently the relevance of the literal that corresponds to it in the ILP setting. We reduce the literals judged irrelevant and obtain a filtered version of the relational example. To filter the whole relational problem, we iterate the process taking *each example in turn.*

3. Multi-instance Propositionalization

We address learning in the non-recursive Datalog language, which is a non recursive clause language without function symbols other than constants. We use the typical learning by implication paradigm (de Raedt, 1997), where examples are clauses and hypotheses are sets of clauses:
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. In this setting, the logical implication is equivalent to θ-subsumption (Gottlob, 1987), which is decidable but NP-complete:

Definition 1 (θ-subsumption (Plotkin, 1970)). A clause C θ-subsumes a clause D iff there exists a substitution θ s.t. $C\theta \subseteq D$

The multi-instance propositionalization (Zucker and Ganascia, 1996, Alphonse and Rouveirol, 2000) is a representation shift that reformulates the FOL learning problem as an attribute-value one, preserving the expressive power of the original problem. This approach 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 introduces a fixed set of attributes - literals of the pattern - to re-describe all the FOL examples. Each matching substitution determines the values of attributes.

To clarify the kinds of values attributes can take, (Zucker and Ganascia, 1998) and (Sebag, 1998) observe that 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 space in which boolean attributes represent the relational part of the propositionalization pattern, and valued attributes its functional part (see (Sebag, 1998) 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 (Sebag and Rouveirol, 1997). 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.

3.1. Propositionalization of the Training Set

We briefly describe the propositionalization process through an example as it is similar to the one defined in (Alphonse and Rouveirol, 2000), althought for another purpose. We define the *pattern of propositionalization* P as the variabilization of a given positive example, under the considered partial order imposed by θ-subsumption.
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 to 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: In the trains problem, inspired by R. Michalski's, let E, and E' be two Datalog clauses representing two positive examples and NE be a Datalog clause describing a negative example of the target concept:

E': train(t) :-	car(t,c1),short(c1),load(c1,l11),rect(l11),
	car(t,c2),short(c2),load(c2,l21),circ(l21).
E: train(t) :-	car(t,c1),long(c1),load(c1,l11),rect(l11),
	car(t,c2),long(c2),
	car(t,c3),short(c3),load(c3,l31),hex(l31).
NE: train(t) :-	car(t,c1),long(c1),load(c1,l11),hex(l11),
	car(t,c2),short(c2),load(c2,l21),circ(l21),
	car(t,c3),long(c3),load(c3,l31),rect(l31).

This relational representation states for example that E' has two short cars, one carrying a rectangular-shaped load and the other one a circular-shaped one.
To re-describe the whole dataset, we use E' to construct the pattern P. It is built as the following variabilization of E' (omitting the head):

$$\begin{array}{lcl} P & : & \text{car(V,W),short(W),load(W,X),rect(X),} \\ & & \text{car(V,Y),short(Y),load(Y,Z),circ(Z).} \end{array}$$

We can observe that since in the structural problem constant names are local within clauses (so called Skolem constants), it does not matter whether we work with skolemized clauses, or variabilized clauses so that links between literals are preserved in the variabilization. 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. Without loss of generality, we use the links between variables to constrain the matching space being searched. The propositionalization process first builds the substitution $\sigma_{E,1} = \{V/t, W/c1, X/l11, Y/c1, Z/l11\}$. Note that the literals $short(W)$, $short(Y)$ and $circ(Z)$ of P have not been matched to any literal of E, hence the value "false" (0). When searching for another possible matching for P's and E's literals, the next valid substitution computed is another matching for $car(V, Y)$: $\sigma_{E,2} = \{V/t, W/c1, X/l11, Y/c2\}$. For this substitution, neither the literals $load(Y, Z)$, $short(W)$, $short(Y)$ and $circ(Z)$ nor the variable Z have been matched. Again, another matching will yield $\sigma_{E,i} = \{V/t, W/c2, Y/c3, Z/l31\}$. In our example, six other substitutions (and therefore six other boolean vectors) are obtained when searching for all possible partial matchings of variables of P with constants of E. □

3.2. *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

Table 1. The tabular representation of a FOL problem

P	car(V,W)	short(W)	load(W,X)	rect(X)	car(V,Y)	short(Y)	load(Y,Z)	circ(Z)
σ_P	1	1	1	1	1	1	1	1
$\sigma_{E,1}$	1	0	1	1	1	0	1	0
$\sigma_{E,2}$	1	0	1	1	1	0	0	0
...								
$\sigma_{E,i}$	1	0	0	0	1	1	1	0
...								
$\sigma_{NE,1}$	1	0	1	0	1	0	1	0
$\sigma_{NE,2}$	1	0	1	0	1	1	1	1
...								
$\sigma_{NE,j}$	1	0	1	1	1	1	1	1
...								

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 this vector and a FOL description where the syntactic representative is simply a subset of the chosen propositionalization pattern. It has to be noted that the interpretation of the boolean vectors differs from the classical "learning from interpretations" paradigm in AVL (de Raedt, 1997). In this setting, the value of the attribute indicates if the attribute is true or false, whereas in the learning by implication paradigm the value of the attribute indicates if the attribute is present or not.

As pointed out in (Zucker and Ganascia, 1996), the new instance space is no longer a set of positive and negative vectors but is now a so-called *multi-instance problem* (MIP[1])(Dietterich et al., 1997), formerly called a multi-part problem: for each positive FOL example, at least one of its associated vectors is positive, and for each negative example, all its associated vectors are negative.

We generalize the equivalence theorem (Zucker and Ganascia, 1996) to our framework:

THEOREM 1 *Given P (a formula in the hypothesis language), searching for a solution which is a subset of P (2^P, up to variable renaming) is equivalent to solving the multi-instance problem, obtained from multi-instance propositionalization with P as pattern.*

4. Feature Selection in ILP

Before describing in detail each step of the filtering approach proposed in Sect. 2 we need to understand the context in which we perform the filtering. In the proposed approach, feature selection is performed by first reformulating the relational problem, and then applying feature subset selection techniques. Consequently, the filter methods used for feature selection must perform well in the new instance space. More precisely, the proposed reformulation of the original relational problem will

inevitably produce noise, and therefore, the issue of noise resistance has to be addressed while designing a filter algorithm. We focus only on the noise artificially generated by the reformulation, and not the noise present in the original data.
To begin with, the multi-instance representation of the problem could be seen as a class-noisy representation of the positive data (Blum and Kalai, 1998, Chevaleyre and Zucker, 2000). Indeed, 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. However, this problem has been well studied in the ILP community and relaxing the completeness of top-down algorithms is typically applied (Zucker and Ganascia, 1996).

Therefore we focus instead on other sources of noise which are more closely related to our approach. In order to illustrate the impact of noise, we upgraded the boolean-attribute filter FOCUS (Almuallim and Dietterich, 1994) (by relaxing completeness) and performed simple experiments on a train-like artificial problem similar to the one in Sect. 3.1. FOCUS uses a top-down exhaustive search in the attribute space and is not noise-resistant. Two datasets have been generated, one with a conjunctive target concept , and the other with a disjunctive one.

Below we discuss the impact of noise due to the change of representation for the two kinds of concepts: the conjunctive and the disjunctive ones.

4.1. The Conjunctive Learning Case

We validate, first, the relevance of the approach developed in the paper, by multi-instance propositionalizing the FOL problem and retaining all vectors, that is, we explored the whole matching space between the pattern and an example. As showed in the first row of Table 2, FOCUS finds all relevant literals and only those literals. This behavior is a consequence of Theorem 1, as in the learning by implication paradigm, the conjunction of the minimal set of relevant attributes is also a solution of the conjunctive concept learning task.

However, in the general case, as pointed out by (Sebag and Rouveirol, 1997), 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 $\prod n_i^{m_i}$ vectors, where n_i and m_i are the number of occurrences of non-determinate literals based on the predicate symbol p_i, in the FOL example e and in P, respectively. For instance, in the mutagenesis dataset, described in Sect. 6, the potential number of matchings is 40^{40}.
This set, however, is highly redundant and only few vectors are sufficient to represent the whole instance space. Indeed, this set is known to be the set of the most specific vectors in the boolean lattice-like instance space: the nearest-hits and nearest-misses of the pattern (Alphonse and Rouveirol, 2000). A trivial upper-bound on its size is the maximal number of incomparable elements in the above instance space which is known to be $\binom{n}{\lfloor n/2 \rfloor}$, with $n = |P|$. Therefore, instead of dealing with a vector space of size exponential in both $|e|$ and $|P|$, we have a space exponential only in $|P|$ (2^{40} instead of 40^{40}). Algorithms can be designed to ap-

proximate this set of non-redundant vectors in order to cope with the intractability of the MIP. 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 approximation of the initial relational problem as *bounded* multi-instance propositionalization. Such 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 "false", introducing attribute noise. The resulting noise in the approximating set of vectors is particular in the sense that it is generated artificially by the bounded MIP, inherent in the proposed paradigm. It is strongly related to the quality of the heuristic used to extract as specific vectors as possible. A natural question arises: as the noise is generated by the approximating algorithm, can the amount of so introduced noise be estimated? Unfortunately, in the general case, when we approximate the non-redundant set, the answer is negative. This is due to a negative result concerning the so-called Max-Compatible-Binary-Constraint-Satisfaction problem (equivalent to the problem of computing the exact set of non-redundant elements) that states that we can not approximate the solution within any constant unless $P = NP$ (Jagota, 93). Further investigation could focus on weakening the representation language in order to lift this limitation in the general case (e.g. k-local (Cohen, 1993)).

For now, we have only investigated a very simple bounded MIP scheme, following the idea of (Sebag and Rouveirol, 1997) of sampling the space of matchings. We apply a stochastic process where k matchings are selected to yield a bounded reformulated problem, with k being a user-supplied parameter. Row 2 in Table 2 shows the FOCUS' performance degrading on the approximated problem, due to attribute-noise.

4.2. The Disjunctive Learning Case

In this case, the concept consists of a number of different subconcepts whose disjunction is the representation of the entire concept. Any two examples of a disjunctive concept may belong to two different subconcepts - in particular the pattern and the positive example being reformulated may not belong to the same sub-concept. That is, some literals may belong to the sub-concept the P corresponds to, and not belong to the sub-concept the example represents. These literals, used as attributes to describe the several reformulations of the example will not be matched. Therefore the MIP will produce irrelevant[2] 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. Third row of Table 2 represents this case of noise due to the disjunctive character of the target concept. We can observe that the performance worsens in both measures.

This last property seems to invalidate any FSS scheme unable 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. This filtering algorithm is not parameterized with a noise level, but removes fewer literals as the level of noise increases. This is exemplified in the last row of Table 2.

Table 2. Filtering on an artificial problem

	relevant literals (%)	mean reduction (%)
conj-exhaustive	100	90
conj-bounded	80	90
disj-bounded 1	25	87.8
disj-bounded 2	100	61

5. A Filtering Algorithm for the Reformulated Problem

Relief has been shown to successfully handle attribute-noise (Kira and Rendell, 1992). 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 otherwise. 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 discussed in (Pompe and Kononenko, 1995) to prune irrelevant branches of refinement graphs. They adapted it to work in the learning by implication paradigm, and although we use a Relief-like filter for a completely different task, we share the same weight update procedure. We refer the reader to (Pompe and Kononenko, 1995) for a detailed description.
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 to one, we can only assess the relevance of attributes of the negative examples which are all more general than the pattern. So, the gain of an attribute wrt a negative vector depends on whether an attribute is set to one or to zero. In our learning setting , if the value of the attribute is one, the attribute 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 2.

For convenience, we can consider the weights normalized between [-1;+1]. If we analyze the two extremal weights, we can see that an attribute occurring 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 vector will get a weight +1, as it is very discriminating.

```
**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 weight is positive.
```

Figure 2. Filtering Algorithm

Before discussing the complexity of our method, we reiterate that in the current realization - due to learning by implication - the pattern P is the bottom-most element of the lattice that represents a bag of instances. Consequently, in this first implementation we are filtering only positive examples, and we use only negative examples in the Relief-like filtering process described above. The time complexity of this algorithm is linear in k, in the number of negative examples and in the complexity of computing one vector (propositionalization). In the case where the sampling scheme described in Sect. 4.1 is used, the total complexity is $\mathcal{O}\left(k\,|E^-|\,|P|\,|e|\right)$. In the current implementation of the paradigm, as we do not use the positive examples, and we process the k negative examples incrementally (one at a time), 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 7.

5.1. *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 3.1. According to section 5, for each negative vector computed, each attribute whose the value is true will have a negative gain. Considering only the vectors given in example 3.1 (as a result of bounded MIP), we have the results given in table 3, 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. : $short(W)$ and $rect(X)$.
Therefore, given the relational problem and the following example as the pattern:

E': train(t) :- car(t,c1),short(c1),load(c1,l11),rect(l11),
car(t,c2),short(c2),load(c2,l21),circ(l21).

The current implementation will produce the following filtered example[3]:

$$E' : train(V) \leftarrow short(W), rect(X).$$

Or equivalently, applying a skolem substitution:

$$E' : train(t) \leftarrow short(c1), rect(l11).$$

Table 3. The output of the filter algorithm

P	car(V,W)	short(W)	load(W,X)	rect(X)	car(V,Y)	short(Y)	load(Y,Z)	circ(Z)
$\sigma_{NE,1}$	-1	+1	-1	+1	-1	+1	-1	+1
$\sigma_{NE,2}$	-1	+1	-1	+1	-1	-1	-1	-1
$\sigma_{NE,j}$	-1	+1	-1	-1	-1	-1	-1	-1
	-1	+1	-1	+0.33	-1	-0.33	-1	-0.33

6. Experiments

In order to validate the proposed paradigm to bring FSS to the realm of ILP, we used its simple instantiation proposed in Sect. 5, and evaluated the training time and the accuracy twice: prior to applying the filter, and after this application.
We considered several learning systems. It is interesting to see that not all state-of-the-art ILP learning systems can take advantage of the filtered example space in a straightforward manner. First of all, Progol and Aleph (in its default setting) perform an exhaustive search in a bounded concept space (namely a A^* search procedure). Therefore, they are going to find the best concept definition containing only relevant literals, according to their quality measure. An effective gain in time can be realized, and it can be traded-off for accuracy by enlarging the bounded hypothesis space, but we leave investigation of this possibility for future research. Secondly, ILP systems based on interpretation or on the closed-world assumption, e.g. Tilde and FOIL, respectively, rely on a complete description of the examples (de Raedt, 1997). Therefore, these systems can not take advantage of the reduced example space. In our experiments, we have used FOIL 6.4 (Quinlan, 1990) with the CWA turned off, and all other parameters with default values, and PROPAL (Alphonse and Rouveirol, 2000), an AQ-like learning system. According to our approach described above, we perform dimensionality reduction on the data, and then feed the result to both learners. We compare their performance on the filtered and unfiltered (original) data. The bounded MIP is performed by sampling the matching space with remplacement, as in Sect. 4.1.

We have evaluated the approach by performing experiments on two domains. In the first domain, known as Mutagenesis (a well-known ILP problems used as a benchmark test), examples consist of structural descriptions of a molecule as a definite clause. The task is to classify the molecules 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 (Srinivasan et al., 1994) for a detailed explanation). 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 (Srinivasan et al., 1994). The accuracy of the learned theory for the

Table 4. Comparison of the performance of FOIL 6.4 and PROPAL on the original datasets and the filtered ones

		FOIL			PROPAL		
		$k = 100$	$k = 500$	unfiltered	$k = 100$	$k = 500$	unfiltered
RF	Accuracy (%)	78.09±0.34	78.25±0.15	75.8	85.54±1.3	85.50±1.73	81.80
	Time (s.)	496±1.5	515±52	4655	69±26	110±1	290
RU	Accuracy (%)	-	-	-	80.71±1.75	82.61±3.73	71.4
	Time (s.)	-	-	-	6.34±1.61	5.96±0.85	10
Receiver	Accuracy (%)	98.5±0.0	98.5±0.0	98.16	98.6 ±0.0	98.6±0.0	97.43
	Time (s.)	< 1	< 1	< 1	< 1	< 1	< 1
Date	Accuracy (%)	98.8±0.0	98.8±0.0	98.6	99.33 ±0.0	99.33±0.0	99.33
	Time (s.)	< 1	< 1	< 1	19.26±0.35	19.36±0.57	21.5
Sender	Accuracy (%)	97.36±0.0	97.36±0.0	97.36	92.74 ±0.0	92.74±0.0	94.24
	Time (s.)	< 1	< 1	< 1	< 1	< 1	2.85

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.

In the DU domain, the task is to recognize parts of documents. Here, we work with three tasks: recognizing the Receiver, the Date, and the Sender. We used the experimental protocol provided in (Esposito et al., 1994). There are some thirty positive and one-hundred and fifty negative examples in each task. The examples contain up to 250 literals. In the FSS literature, a filtering task involving 250 features is not considered a trivial one.

Table 4 compares the performance of FOIL and Propal on the filtered and not filtered datasets. In order to include the filter in the validation process, we filtered only instances used for training, for all validation procedures. Due to the stochastic process of the bounded propositionalization used, each accuracy and time have been averaged over 10 runs; the standard deviation is given.

In the mutagenesis domain, as expected, the performance of the two learning systems 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 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. Also, not only FOIL's accuracy has slightly improved, but its running time has decreased almost by a factor of 10. The deterioration of the performance for the case with $k = 500$ is most likely due to the increase of noise by working with a larger sample of the noisy instance space.
In the DU domain, instead of improving an already high performance on the original data, the proposed method correctly preserves literals whose removal would degrade performance. The results may even improve slightly, while the PROPAL execution

time for the two more difficult tasks decreases (it is halved for the last task, Sender, retaining the accuracy on the original data).

The values of k in these experiments are arbitrary. A judicious choice of k needs to be studied further.

7. 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, following (Sebag and Rouveirol, 1997), 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, like in KBG (Bisson, 1991). 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 these applications. Thirdly, with the upgrade of feature selection techniques - from AVL to FOL - 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 than P to cope with the class-noise issue. 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 [4]. However, the current setting has its advantages due to its low complexity, and its constant space requirement which allows arbitrarily large sample size (here, k in sect. 4.1).

Finally, as we work in the learning by implication paradigm, the roles played by positive and negative examples are not symmetric. For that reason we did not investigate filtering negative examples, i.e. we have just applied the filtering process on each positive example in turn. The extension of the filtering process to negative examples is an interesting follow-up, but the filtering algorithm must differ from the current method and is beyond the scope of this paper.

8. Conclusion

FOL problems typically have to cope with severer dimensionality problems than those in AVL learning. We have presented here the first paradigm to apply, in the ILP context, the feature subset selection approach to reduce the dimensionality of the example space, in languages at least as expressive as non-recursive Datalog clauses. It bridges the gap between the flexible representation of instances in the form of FOL clauses, and the fixed-format requirement of the FSS algorithms. The main idea is to consider filtering one relational example at a time. This uses its literals as a fixed set of attributes for approximating the relational problem by a multi-instance problem. Filtering the MIP allows us to assess the relevance of the example's literals. To filter the whole relational problem, we consider each example in turn. In this paradigm, the feature selection techniques work as a front end filter, applied prior to further processing and to the model building. Moreover, our method is applicable to any partial ordering of the learning space.

In order to empirically evaluate the relevance of the approach, we have proposed a simple instantiation and performed experiments on several tasks from two real-world domains: Mutagenesis, a biochemical domain, considered benchmark in ILP, and the Document Understanding domain. The preliminary results are very encouraging, showing, as expected, an improvement both in time and accuracy, demonstrating the applicability of FSS in ILP.

Our approach is based on the approximation of the original ILP problem by a bounded MIP. A consequence of the approximation is the introduction of noise in the data. As far as we know no research has been done on noise-resistant MIP filters, and any development in this area will be very valuable for ILP.
The noise introduced by the change of representation heavily depends on the quality of the approximation. Unfortunately, as discussed in Sect. 4.1, there is no algorithm that in polynomial time would guarantee the quality of the approximation. Therefore we plan to further investigate stochastic sampling techniques, e.g. GRASP (Feo and Resende, 1995). An interesting complementary approach would be to investigate biases providing either polynomial-space complexity of MIP (e.g. k-local (Cohen, 1993), constrained DHDB (Lavrač and Džeroski, 1994)) or efficient approximation schemes. We expect our method to perform well in these constrained languages.

In this initial investigation, the tasks on which we have experimented represent state-of-the art ILP applications. We plan to further investigate the validity of the approach on problems presented by the KDD community still considered out of scope of current ILP techniques.

Acknowledgments

We are grateful to anonymous ISMIS 2002 and ICML 2002 referees whose comments allowed us to improve the quality of the paper. We thank Michèle Sebag and Céline Rouveirol for discussing their work with us. We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada and of the Computing and Information Technologies Ontario

Notes

1. In this paper, we do not distinguish between the multi-instance propositionnalisation and the multi-instance problem obtained from it

2. This case is exemplified by the FOL learning problem described in Sect. 3.1. 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 $\sigma_{NE,j}$, and therefore should really be negative in our learning setting.
3. This representation of the final example depends of the language of representation used. In this case the language is limited to linked clauses, and the missing predicates linking W and X to the head should be added but not used as refinements.
4. In the current work we have removed the latter limitation. We have also designed a new feature selection technique that takes into account the noise in the data, inherent in the bounded MIP.

References

H. Almuallim and T. G. Dietterich. Learning Boolean concepts in the presence of many irrelevant features. *Artificial Intelligence*, 69(1–2):279–305, 1994.

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.

G. Bisson. KBG: A generator of knowledge bases. In Y. Kodratoff, editor, *Proceedings of the European Working Session on Learning : Machine Learning (EWSL-91)*, volume 482 of *LNAI*, pages 137–137, Porto, Portugal, Mar. 1991. Springer Verlag. ISBN 3-540-53816-X.

A. Blum and A. Kalai. A note on learning from multiple-instance examples. *Machine Learning*, 30:23–29, 1998.

Y. Chevaleyre and J. Zucker. Noise-tolerant rule induction for multi-instance data. In *ICML 2000, Workshop on Attribute-Value and Relational Learning*, 2000.

W. W. Cohen. Learnability of restricted logic programs. In S. Muggleton, editor, *Proceedings of the 3rd International Workshop on Inductive Logic Programming*, pages 41–72. J. Stefan Institute, 1993.

W. W. Cohen. Learning to classify English text with ILP methods. In L. D. Raedt, editor, *Advances in inductive logic programming*, pages 124–143. IOS Press, Amsterdam, NL, 1995.

D. Conklin. Machine discovery of protein motifs. *Machine Learning*, 21:125–150, 1995.

L. de Raedt. Logical settings for concept-learning. *Artificial Intelligence*, 95(1):187–201, 1997.

T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1–2):31–71, 1997.

F. Esposito, D. Malerba, and G. Semeraro. Multistrategy learning for document recognition. *Applied Artificial Intelligence*, 8:33–84, 1994.

T. A. Feo and M. G. C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.

J. Fürnkranz. Dimensionality reduction in ILP: A call to arms. In L. de Raedt and S. Muggleton, editors, *Proceedings of the IJCAI-97 Workshop on Frontiers of Inductive Logic Programming*, pages 81–86, Aug. 1997.

G. Gottlob. Subsumption and implication. *Information Processing Letters*, 24(2):109–111, 1987.

A. Jagota. Constraint satisfaction and maximum clique. In *Working Notes, AAAI Spring Symposium on AI and NP-hard Problems*, pages 92–97. Stanford University, 93.

J.-U. Kietz, U. Reimer, and M. Staudt. Mining insurance data at swiss life. In *The VLDB Journal*, pages 562–566, 1997.

K. Kira and L. A. Rendell. A practical approach to feature selection. In *Proc. of the Ninth Int. Conference on Machine Learning.*, pages 249–256. MK, 1992.

R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1–2): 273–323, 1997.

N. Lavrač and S. Džeroski. *Inductive Logic Programming : techniques and Applications.* Ellis Horwood, 1994.

Y. Lee, B. G. Buchanan, and J. M. Aronis. Knowledge-based learning in exploratory science: learning rules to predict rodent carcinogenicity. *Machine Learning*, 30:217–240, 1998.

H. Liu and H. Motoda. *Feature Extraction, Construction and Selection: A Data Mining Perspective.* Kluwer Academic Publisher, 1998.

C. Nédellec, C. Rouveirol, H. Ade, F. Bergadano, and B. Tausend. *Advances in Inductive Logic Programming*, chapter Declarative Bias in Inductive Logic Programming, pages 82–103. Raedt de L. (ed.), IOS Press, 1996.

G. Plotkin. A note on inductive generalization. In *Machine Intelligence*, volume 5. Edinburgh University Press, 1970.

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.

J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5(3):239–266, 1990.

C. Rouveirol. Expressiveness/Efficiency: the dilemma of the Inductive Logic Programming. LRI, Université Paris, décembre 2000. HDR, in french.

M. Sebag. Resource bounded induction and deduction in fol. In *Proceedings of 4th International Workshop on Multistrategy Learning*, pages 95–105, 1998.

M. Sebag and C. Rouveirol. Constraint inductive logic programming. In L. De Raedt, editor, *Advances In Inductive Logic Programming*, pages 277–294. IOS Press, 1996.

M. Sebag and C. Rouveirol. Tractable induction and classification in first order logic via stochastic matching. In *15th Int. Join Conf. on Artificial Intelligence (IJCAI'97)*, pages 888–893. Morgan Kaufmann, 1997.

G. Silverstein and M. J. Pazzani. Relational cliches: Constraining constructive induction during relational learning. In L. Birnbaum and G. Collins, editors, *Proceedings of the 8th International Workshop on Machine Learning*, pages 203–207. Morgan Kaufmann, 1991.

A. Srinivasan, S. Muggleton, R. King, and M. 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.

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.

J.-D. Zucker and J.-G. Ganascia. Learning structural indeterminate clauses. In D. Page, editor, *Proc. of the 8th International Workshop on Inductive Logic Programming*, pages 235–244. Springer Verlag, 1998.