Relational Learning from Ambiguous Examples

Dominique Bouthinon, Henry Soldano

Relational Learning from Ambiguous Examples

Dominique Bouthinon Henry Soldano

LIPN, CNRS (UMR 7030) Universite Paris13,
Sorbonne Paris Cité F-93430, Villetaneuse, France
{dominique.bouthinon,henry.soldano}@lipn.univ-paris13.fr

Résumé

Dans cet article nous étudions une situation d'apprentissage relationnel où chaque exemple est ambigu dans le sens où il est dissimulé au sein d'un ensemble de descriptions. Ce cas apparait en apprentissage de règles lorsque les valeurs de vérité de certains atomes décrivant un exemple sont inconnues, alors qu'on dispose d'une théorie du domaine permettant de limiter les descriptions possibles de cet exemple. Le but de la tâche d'apprentissage étudiée ici est de trouver, malgré l'ambiguité des exemples, la plus simple représentation du concept sous-jacent. Nous présentons tout d'abord un cadre d'apprentissage de règles du premier ordre à partir d'exemples ambigus, puis une implementation de ce cadre appelée LEAR. Enfin, nous proposons des résultats expérimentaux où LEAR fait face à des degrés d'ambiguité croissants.

Mots Clef

Apprentissage relationnel supervisé, incertitude

Abstract

We investigate here relational concept learning from examples when we only have a partial information regarding these examples, i.e. for each such ambiguous example, we only know a set of its possible complete descriptions, denoted as possibilities. A typical such situation is rule learning when truth values of some atoms are missing in the example description while we benefit from a background knowledge. The learning task investigated here is to find simple representations of the concepts we learn. We first propose an adaptation of rule learning from ambiguous examples. Then, we describe LEAR, an implementation of this setting. At least we discuss various experiments in which we observe how LEAR copes with increasing degrees of incompleteness.

Keywords

Relational supervised learning, uncertainty.

1 Introduction

We consider here supervised relational learning when data representation is (very) incomplete. Due, in particular, to the development of the semantic web, AI is more and more interested in relational data, i.e. data whose natural representation includes relations between objects of various types. Such data are associated with models of knowledge, as ontologies, to help their interpretation and therefore their automatic processing. In this context, data is also naturally partially known: when gathering observations for some task most data represents only partial descriptions. We propose here an approach of supervised relational learning extensively using background knowledge to deal with high level of incompleteness in data representation. The present work relies on two ideas. The first one is that by properly using background knowledge, we can extract all relevant information inside incomplete observations, therefore being able to learn even from very incomplete data. Various methods, except *abductive induction* we refer to later in the related works section, handle such *missing data* but does not plainly use background knowledge. The second idea is that our purpose is not to find a best classifier given the incomplete observations we expect to encounter in the future, but rather to find the simplest representations of the underlying concepts we learn, i.e. representations we would learn should the observations be complete. Our purpose is therefore the discovery of correct and simple representations and not building accurate classifiers for incomplete observations. As a consequence, in our experiments we will learn from such incomplete examples but compute the accuracy of the resulting hypotheses on complete examples.

We illustrate hereunder the latter idea on a very simple example. Consider a world of birds from which we want to learn the unary predicate *fly*. Any bird is described using the binary predicates $\{color, lives\}$. They are two colors, *red* and *green* (some birds can be both *red* and *green*, some have no color). They are two known continents: *europe* and *africa* (some birds can live in the two continents, some live neither in Europe nor in Africa). A complete example is a clause whose the head is an instance of the target concept, and the body contains all the true atoms relative to the example, for instance: $fly(t) \leftarrow color(t, red) \wedge lives(t, europe)$. Implicitly this clause stipulates that $color(t, green)$ and $lives(t, africa)$ are false : the body of a clause is an Herbrand interpretation of the language used to describe the birds.

Now suppose the only thing we know about t is that it is *red* and not *green*. From this partial information we can build four possible complete examples :
x_1 = *fly*$(t) \leftarrow color(t, red)$,
x_2 = *fly*$(t) \leftarrow color(t, red) \wedge lives(t, europe)$,
x_3 = *fly*$(t) \leftarrow color(t, red) \wedge lives(t, africa)$ and
x_4 = *fly*$(t) \leftarrow color(t, red) \wedge lives(t, europe) \wedge lives(t, africa)$.
The set $e = \{x_1, x_2, x_3, x_4\}$ is called an *ambiguous example* containing four *possibilities*. A single possibility of an ambiguous example (here x_2) is the actual complete example that would has been observed if there was no ambiguity. In the following we will simply call x this hidden actual complete example. Consider a hypothesis H = *fly*$(X) \leftarrow lives(X, africa)$ (we allow constant in hypotheses). Let us assume that e is labelled positive, so x is a positive instance of the target concept *fly*. One wonders whether H covers the complete example x hidden in e. A sceptical view consists in answering *no*: we cannot guarantee that H covers x[1] because H does not cover all the possibilities of e. A credulous view answers *yes*: H possibility covers x because it covers possibilities of e.
Were the purpose to build a classifier to be used in observations as ambiguous as the one we observe, we should be sceptical, in order to guarantee that each actual positive complete example would be covered by our final hypothesis. This is the approach used by the relational learner TILDE [1]. But as our purpose is to find the correct representation of the target concept, then we have to be credulous: if we have enough ambiguous examples we can eliminate most of incorrect hypotheses and return a hypothesis that can't be rejected when considering our set of ambiguous examples. This idea was informally proposed in [4] as an extension of Mitchell's version space definition. Here, for instance the hypothesis $fly(X) \leftarrow color(X, green)$ would be rejected: it cannot cover the positive instance x because it covers no possibility of e. Note that the same idea applies to negative instances: here being credulous means we keep in the solution space the hypotheses that do not cover at least one possibility of the negative ambiguous examples. Equivalently we reject the hypotheses that cover all the possibilities of a negative ambiguous example.
Overall this means that learning relies here on a *compatibility* relation. An example is compatible with a hypothesis when what we know does not exclude this hypothesis from the version space. However, the compatibility relation depends on the label, either positive or negative, of the example.
Let us illustrate now the former idea mentioned above, namely plainly using background knowledge when considering whether an ambiguous example is compatible or not with a given hypothesis. Going back to our previous example, suppose we also know that, whatever is the observation, its hidden actual complete example should satisfy the following background knowledge $B = \{\leftarrow lives(X, europe) \wedge lives(X, africa), lives(X, europe) \vee lives(X, africa)\}$ [2]. This new information stipulates that a bird lives either in Europe or in Africa. This limits the ambiguity, for instance it allows us to discard the possibilities x_1 and x_4 of the previous example e.

Consider now another bird, an ostrich, called o which is a negative instance of the concept *fly*. We know that o either is red and lives in Africa, or is green and lives in Europe. Then o is represented as the ambiguous example $e^- = \{y_1, y_2\}$ with
y_1 = *fly*$(o) \leftarrow color(o, red) \wedge lives(o, africa)$ and
y_2 = *fly*$(o) \leftarrow color(o, green) \wedge lives(o, europe)$.

Consider a hypothesis having two clauses: $H = \{$*fly*$(X) \leftarrow lives(X, europe)$, *fly*$(X) \leftarrow color(X, green)\}$. Such a multiple hypothesis covers a (complete) example whenever one of its member covers the example, meaning that the underlying concept can be satisfied in two different ways. Then H is compatible with a positive ambiguous example e if a clause of H covers a possibility of e. And H is compatible with a negative ambiguous example e when a possibility of e is covered by no clause of H. This is the case of the possibility y_1 of the example e^-.

Whenever the heads of the clauses are not relevant, the above informal presentation of concept learning from ambiguous examples relies on a logical setting denoted as *learning from interpretations*. For instance, assume the target concept *fly* (there is no variable), then a hypothesis clause *fly* $\leftarrow color(X, green)$ covers an example clause *fly* $\leftarrow color(t, green)$ when $\{color(t, green)\}$ is a Herbrand model of $color(X, green)$.

The learning from ambiguous examples framework also relies on the framework of *learning from entailment*: H covers an ambiguous example e if a clause of H *subsumes* (or entails) a clause of e (see [3] for a discussion on the various logical learning settings).

In [2] we have investigated the effect of incompleteness in the propositional concept learning from interpretations setting. To handle ambiguity we proposed a concept learning program, called *Lea* based on the ideas mentioned above and using compatibility to evaluate hypotheses.

LEAR is a relational learner that, given a learning set of incomplete positive and negative examples of a target concept together with some background knowledge, selects a hypothesis compatible with the examples of the learning set. LEAR first builds a representation of each ambiguous example as a set of possibilities, and then uses a standard greedy set covering strategy to build a hypothesis.

Section 2 introduces the learning from ambiguous examples setting, whereas LEAR is described section 3. The results of various experimentations are shown section 4, followed by the related works and the conclusion sections 5 and 6.

[1] in the sense that H *θ-subsumes* x: there exists a substitution θ with $H\theta \subseteq x$.

[2] $\leftarrow a \wedge b$ is a writing for the clause $\neg a \vee \neg b$.

2 Learning first order rules from ambiguous examples

In our setting, hypotheses and ambiguous examples are expressed from a first order language whose the set of ground atoms is HB. All examples are described using the same predicates, however in the basis HB the number of constants, and therefore the size of the basis depend on the example. However in what follows, with no loss of generality, we consider all these bases as identical. Our general purpose is to learn a set of clauses $H = \{t_1 \leftarrow h_1, \ldots, t_n \leftarrow h_n\}$, where each t_i is built on the same target predicate t, and which is *compatible* with a set of ambiguous examples E. An ambiguous example is a set of ground clauses concluding on a ground instance t_e of t.

2.1 The compatibility relations

We use the following compatibility relations between sets of clauses :

Definition 1 *Let H be a hypothesis and e be an ambiguous example. H is a set of clauses and e is a set of grounded clauses:*

- *H is compatible$^+$with e iff at least one clause of e is θ-subsumed by a clause of H,*
- *H is compatible$^-$with e iff at least one clause of e is θ-subsumed by no clause of H.*

Example 1 *Consider a world containing two objects a and b, each one can be either a rectangle or a triangle, possibility related by the relation $on(X,Y)$. The goal is to learn a particular configuration of these two objects represented by the target concept $gc(X,Y)$. Consider the following ambiguous examples:*
$e^+ = \{d_1, d_2\} = \{gc(a,b) \leftarrow rectangle(a) \wedge rectangle(b) \wedge on(a,b),\ gc(a,b) \leftarrow triangle(a) \wedge rectangle(b) \wedge on(a,b)\}$ *and*
$e^- = \{d_2, d_3\} = \{gc(a,b) \leftarrow triangle(a) \wedge rectangle(b) \wedge on(a,b), gc(a,b) \leftarrow triangle(a) \wedge triangle(b) \wedge on(a,b)\}$.
Let H be the set of clauses $\{c_1, c_2\} = \{gc(X,Y) \leftarrow triangle(X) \wedge rectangle(Y) \wedge on(X,Y)\ ,\ gc(X,Y) \leftarrow rectangle(X) \wedge triangle(Y) \wedge on(X,Y)\}$. H is compatible$^+$ with e^+ (d_2 is θ-subsumed by c_1) and compatible$^-$with e^- (d_3 is θ-subsumed neither by c_1 nor by c_2).

2.2 Discarding useless possibilities

We have shown in [2] that there were no need to represent all the interpretations contained in an ambiguous examples. We extend this result to our clausal framework. Let e be an ambiguous example, then $min(e)$ $(max(e))$ is the set of minimal (maximal) clauses with respect to the inclusion order on grounded clause. We have then the following result:

Proposition 1 *Let H be a hypothesis and e be an ambiguous example then:*

- *H is compatible$^+$with e iff H is compatible$^+$with $max(e)$,*
- *H is compatible$^-$with e iff H is compatible$^-$with $min(e)$.*

Proof 1 *We give hereunder the proof regarding the compatible$^+$relation. The proof regarding the compatible$^-$relation is quite similar.*

$\Rightarrow$ *H is compatible$^+$ with e means that there is some clause c of H, a clause d of e, and a substitution θ such that $c\theta \subseteq d$. Furthermore, d is included in (or equal to) at least one clause $d_{max} \in max(e)$. Then $c\theta \subseteq d_{max}$, and therefore H is compatible$^+$with $max(e)$.*

$\Leftarrow$ *H compatible$^+ max(e)$ means that a clause of H θ-subsumes a clause of $max(e) \subseteq e$, so H is compatible$^+ e$.*

This means that we only need the maximal ground clauses of each positive ambiguous example, and the minimal ground clauses of each negative ambiguous example. This is important when the background knowledge is a Horn clausal theory. In such a case each negative ambiguous example has one single minimal ground clause and therefore can be represented as one possibility[3]. However, in our experiments we also consider ambiguous examples built from non Horn clausal theory, i.e. including, for instance, a clause as $\text{dog}(X) \vee \text{wolf}(X) \leftarrow \text{observed}(X)$ meaning that the animals we consider are either dogs or wolves.

2.3 Multi-table representation

To save space we have introduced a multi-table representation of ambiguous examples. The key idea is to part the bodies of the clauses describing an ambiguous example in parts, called *tables*, such that the set of these bodies can be written as a Cartesian product of these tables, as in the following propositional example (first order examples exactly follow the same principle):

Example 2 *Let $e = \{t \leftarrow a \wedge c,\ t \leftarrow a \wedge d,\ t \leftarrow b \wedge c,\ t \leftarrow b \wedge d\}$. The set of bodies of the clauses of e is $T(e) = \{\{a,c\},\{a,d\},\{b,c\},\{b,d\}\}$. Note that $T(e) = T_1 \times T_2 = \{\{a\},\{b\}\} \times \{\{c\},\{d\}\}$. Finally $(t, T_1 \times T_2)$ is the multi-table representation of e.*

Let us sketch the way we find the tables of an ambiguous example through our propositional example (the same method is used in the relational case). An ambiguous example is first intentionally represented as a clausal theory. Consider the set of atoms $HB = \{a,b,c,d\}$ used to describe the above mentioned ambiguous example e. The intentional representation of e is $int(e) = \{a \vee b, \neg a \vee \neg b, c \vee$

[3] A Horn clausal model has a unique herbrand minimal model.

$d, \neg c \vee \neg d\}$. Note that the bodies of the clauses in e are the Herbrand models of the clausal theory $int(e)$.
To compute the tables we first build a graph whose vertices are the atoms of HB. In the graph an edge links two atoms whenever they belong to a common clause in $int(e)$. The connected components of this graph define a partition on HB. In our example there are two connected components $HB_1 = \{a, b\}$ and $HB_2 = \{c, d\}$. This partition induces a partition of $int(e)$ in two clausal theories $int(e)_1 = \{a \vee b, \neg a \vee \neg b\}$ and $int(e)_2 = \{c \vee d, \neg c \vee \neg d\}$, whose atoms are in HB_1 and HB_2 respectively (see [2] for the proofs supporting this partition). At last table T_1 (T_2) contains the Herbrand models of $int(e)_1$ ($int(e)_2$). Let us point that the multi-table representation can lead to an exponential gain on space.

3 LEAR

LEAR is a swi-prolog ([10]) implementation of the above mentioned setting of learning from ambiguous examples. It learns a set of first order clauses $H = \{t_1 \leftarrow h_1, \ldots, t_n \leftarrow h_n\}$ compatible with a set of ambiguous examples $E^+ \cup E^-$.

3.1 The algorithm

LEAR is a greedy set covering algorithm. In its basic form, its inputs are a set $E^+ \cup E^-$ of positive and negative ambiguous examples together with a parameter W representing the size of the beam involved in the beam search performed by the *bestClause* function (see [2]). Starting from the clausal theory $H = \emptyset$, it iteratively adds a clause c to H, where c is *compatible*$^+$with a part of E^+ and *compatible*$^-$with E^-. Then we remove from E^+ the positive examples *compatible*$^+$with the current hypothesis H, and discard newly inconsistent possibilities from each negative ambiguous examples in E^- (see below). Except this last step, and considering that we use here compatibility rather than covering, this is a standard Top-Down rule learner. The *bestClause* function [4] uses a beam search to find the next clause to add to the current solution H. It starts choosing a *seed*, that is an ambiguous positive example, to restrict the space of clauses in the same way as PROGOL [7]. Then the set of candidates C is initialized with the single clause $t \leftarrow$. A beam search is then performed. At each step of the search, the refinement operator $\rho(C, seed)$ returns S, the maximally general specializations of the clauses in C that are *compatible*$^+$ with the *seed*. If some clauses in S are better than the current $best$ clause, a new $best$ clause is defined. Then the W clauses of S displaying the best ratio of compatible examples over $E^+ \cup E^-$ (i.e the best *scores*) are selected while exact *compatible*$^-$clauses are pruned. This process continues until C is empty or all clauses can be pruned from C: no clause in C can be refined with a better evaluation than the current best solution. Note that the returned clause is necessarily *compatible*$^+$with at least the positive ambiguous example chosen as seed.

[4] LEAR and *bestClause* are similar to the equivalent algorithms described in [2] except that they deal with first order clauses.

3.2 The compatibility relations

Let $e = \{t_e \leftarrow p_1, \ldots, t_e \leftarrow p_n\}$ be an ambiguous example and c be a clause To check whether c is *compatible*$^+$ with e we verify that c *θ-subsumes* a possibility of e. To check whether c is *compatible*$^-$with e we verify that there exists a possibility of e that is not subsumed by c.
However, as the solution set $H = \{c_1, \ldots, c_n\}$ is built by incrementally adding clauses, we must ensure that H is *compatible*$^-$as a whole with the negative examples. This entails to check that all the clauses in H rely on at least one same possibility of each negative example. For instance, consider a negative ambiguous example $e = \{t \leftarrow a, t \leftarrow b\}$ and assume $H = \{t \leftarrow b\}$. Then H is *compatible*$^-$with e by the possibility $t \leftarrow a$ which is not subsumed by the single clause in H. Suppose now $t \leftarrow a$ is a candidate clause to be added to H. We see that $\{t \leftarrow a\}$ is *compatible*$^-$with e by the clause $t \leftarrow b$. But $H = \{t \leftarrow b, t \leftarrow a\}$ would not be *compatible*$^-$with e because the two clauses in H do not rely on the same possibility of e to ensure the *compatibility*$^-$.

4 Experiments

LEAR has been experimented on an artificial problem from the Bongard domain and on the UCI student loan problem. In all the experiments the concept is learned from ambiguous examples and its accuracy is evaluated on a separated test set of complete examples. The results of LEAR are compared to those of TILDE [1] that has to learn from non ambiguous examples.
We consider that examples are drawn following some distribution and that each example is then submitted to a masking process, according to some ambiguity level p, and results in an ambiguous example, representing all the possible descriptions of the complete example. This ambiguous example can be used as it is by LEAR. A corresponding non ambiguous example has to be produced, at the price of loosing some information, to be used by TILDE. We detail hereunder the generating process.

4.1 Methodology

Building the ambiguous examples. Each complete example has a label l either positive or negative, and is described as a clause $x = t \leftarrow b$, where b is a ground conjunction and t is a ground instantiation of the target concept. In addition we are provided by a clausal theory B representing the background knowledge we have about the clause body. Because x is complete, b is a conjunction whose the equivalent set of atoms is a Herbrand model of B. The set of possibilities issued from a complete example is built as follows:

a) Each atom of the body b is discarded with some fixed probability p. We obtain then a smaller conjunction $b' \subseteq b$.

b) We build the set of Herbrand models of the resulting clausal theory $b' \wedge B$. In such a set $\{b_1, \ldots, b_n\}$ each b_i is an alternative complete body of the initial clause and $e = \{t \leftarrow b_1, \ldots, t \leftarrow b_n\}$ is the ambiguous example derived from $t \leftarrow b'$: it contains all possible complete examples knowing B and b'.

According to Proposition 1 LEAR considers only maximal clauses in $\max(e)$ when e is a positive example and minimal clauses in $\min(e)$ when e is a negative example.

The standard way to cope with missing data in TILDE is to assume that all unknown ground atoms are false. Using the background knowledge B makes possible to complete the clause $t \leftarrow b'$, by adding to b' atoms that can be deduced from B and b'. This is performed by turning the clause into $t \leftarrow b''$ where b'' is the intersection of all clauses in $min(e)$.

Depending of the nature of the background knowledge B there are various cases:

- B is empty or only made of definite clauses. In this case, $\mid \min(e) \mid = \mid \max(e) \mid = 1$.
- B is a Horn clausal theory, i.e contains also headless clauses representing integrity constraints as $\leftarrow square(X) \wedge circle(X)$. In this case $\mid \min(e) \mid = 1$ whereas $\mid \max(e) \mid \geq 1$.
- B is a normal clausal theory, i.e also contains clauses as $square(X) \vee circle(X) \leftarrow shape(X)$, and both sets of minimal and maximal clauses have size greater than 1.

When comparing the information about examples given to LEAR and TILDE, LEAR is given the whole information about each ambiguous example e and will be credulous, retaining any hypothesis which is compatible with e. On contrary TILDE is given a unique clause c (the above mentioned $t \leftarrow b''$) and therefore looses some information when B contains normal clauses. Note that c is included in all the possibilities of e, so it represents what is certainly true in e. Therefore, TILDE is credulous regarding negative examples, not covering c implies not covering at least one possibility of e, but sceptical regarding positive ones: covering c implies covering all the possibilities of e.

This means that TILDE performs a cautious (skeptical) covering of incomplete positive examples resulting in more branches in the logical tree than necessary to classify complete examples. Therefore, the comparisons in the experiments hereunder are not perfectly fair, as the two methodologies have different purposes. TILDE builds a tree which purpose is to classify incomplete examples as those in the learning set, whereas LEAR searches for a simple representation of the underlying concept and is therefore only intended to classify complete examples. Still, the accuracies on the complete test examples are comparable.

Evaluation. The evaluation is performed through a 10-10 cross-validation process, applied to each level of ambiguity).

In the artificial case, N examples are freely drawn from the distribution and the masking process is applied to each of them. As a result we obtain two sets of equal size N: E containing the original complete examples and E^a containing the ambiguous versions of these examples. A cross-validation step consists in considering as the test set the fold $E_i \subset E$ and as the learning set $E^a - E^a_i$, where E^a_i is the subset of E_a corresponding to E_i.

In the UCI problems case, as we only have a limited sample E of examples, from each fold E_i containing N_i complete examples, we build an ambiguous fold E^a_i of $10 * N_i$ ambiguous examples by drawing and masking examples from E_i. We obtain then E^a as the union of these folds and proceeds as in the artificial case.

4.2 Experiments

Bongard (Artificial). In the Bongard domain [6] examples are diagrams made of various geometrical objects. In this experiment each example is made of 4 objets, and satisfies the following background knowledge: $\leftarrow in(X,X), \leftarrow in(X,Y) \wedge in(Y,X), \leftarrow triangle(X) \wedge square(X), \leftarrow triangle(X) \wedge circle(X), \leftarrow square(X) \wedge circle(X), \leftarrow config(X,up) \wedge config(X,down), \leftarrow config(X,Y) \wedge circle(X), \leftarrow config(X,Y) \wedge square(X), square(X) \vee circle(X) \vee triangle(X)$.

The target concept can be represented as the three following clauses:
$bongard \leftarrow config(X,up) \wedge in(X,Y) \wedge circle(Y)$,
$bongard \leftarrow config(X,Y) \wedge in(Y,Z) \wedge triangle(Z)$,
$bongard \leftarrow in(X,Y) \wedge circle(Y) \wedge square(X) \wedge config(Z,C)$.

We have run experiments with ambiguity levels ranging from $p = 0$ to $p = 0.9$ and with an increasing number of examples $N = 1000, 2000, 4000, 8000, 16000$.

In this expriment we want to check the expected convergence of LEAR at any ambiguity level provided that there is enough learning examples.

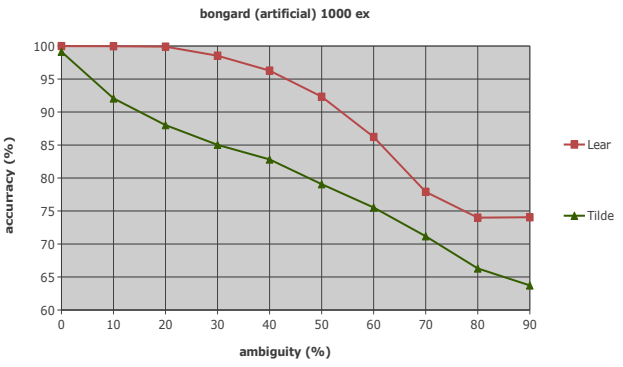

Figure 1: Accuracies on the *Bongard* (artificial) dataset (1000 ambiguous examples)

LEAR										
N	0	10	20	30	40	50	60	70	80	90
1000	100	100	98	98	96	86	89	78	74	74
2000	100	100	100	100	98	95	91	83	75	73
4000	100	100	100	100	99	98	94	88	76	74
8000	100	100	100	100	100	99	96	92	78	76
16000	100	100	100	100	100	99	97	94	79	75
TILDE										
N	0	10	20	30	40	50	60	70	80	90
1000	99	92	88	85	83	79	76	71	66	64
2000	100	94	90	87	84	82	80	76	70	66
4000	100	95	91	88	85	82	80	77	72	66
8000	100	95	92	88	85	82	80	78	73	66
16000	100	96	92	89	85	82	80	78	75	69

Table 1: Accuracies of LEAR and TILDE on the *bongard (artificial)* problem, with $N = 1000$, 2000 4000, 8000, 16000 at ambiguity levels ranging from 0% to 90%.

Figure 1 displays the accuracies of LEAR and TILDE when learning at various accuracy levels from 1000 ambiguous examples. Clearly LEAR outperforms TILDE benefiting from its credulous/credulous bias and the related used of background knowledge, at least for low and moderate ambiguity levels. Table 1 investigates the dynamics: what is the benefit of having an increasing number of examples ? Clearly, LEAR, as expected, benefits from processing more examples, the convergence is slow but it reaches 100% accuracy at ambiguity level 40% when learning from 16000 examples. Tilde also benefits form the increasing number of examples, but, because of its skeptical bias regarding positive examples its accuracy reaches, at each ambiguity level an asymptotic value. For instance, at ambiguity level 40%, TIILDE reaches a 85% accuracy with 4000 examples, and does not increase this accuracy even when learning from 16000 examples.

Loan (UCI). The *Loan* problem consists in classifying bank loans into good and bad loans. As described in [8], the dataset displays 8 relations and contains, for each loan, both customer information and account information. In this problem you did not use any background knowledge, therefore a single possibility is enough to represent all possible complete examples of each ambiguous example. The initial set of complete examples contained about 1000 examples from which we have built about respectively 1000, 5000 and 15000 ambiguous examples.

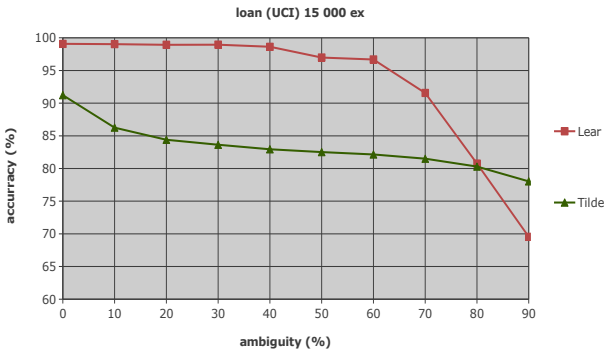

Figure 2: Accuracies on the *Loan* dataset (15000 ambiguous examples)

ambiguity	system	1000	5000	15000
0%	LEAR	96	99	99
	TILDE	90	91	91
10%	LEAR	94	99	99
	TILDE	87	88	86
20%	LEAR	91	99	99
	TILDE	85	85	84
30%	LEAR	87	97	99
	TILDE	84	84	84
40%	LEAR	84	95	99
	TILDE	82	83	83
50%	LEAR	82	92	97
	TILDE	81	82	83
60%	LEAR	77	89	97
	TILDE	78	82	82
70%	LEAR	74	82	92
	TILDE	77	81	81
80%	LEAR	66	75	81
	TILDE	75	78	80
90%	LEAR	62	65	70
	TILDE	71	77	78

Table 2: Accuracies of LEAR and TILDE on the *Loan* problem, considering 1000, 5000 and 15000 ambiguous examples.

Table 2 and Figure 2 show that LEAR have much higher accuracies than TILDE on this problem until level 50%, then the accuracies of LEAR strongly decrease until both programs display similar accuracies.

Regarding the running time, building the extensional ambiguous examples from the facts and the background knowledge is sometimes expensive but tractable. In all our experiments this building time never exceeded 20 mn for a whole dataset whatever was the level of ambiguity. Learning was in general much faster. However, learning can be slowed down when the clauses in the current solution are made of many atoms.

5 Related work

Regarding first order representations, incompleteness, as defined here, has been mainly addressed in works concerning induction and abduction. in the framework of Inductive Logic Programming (ILP). Brave induction [9] addresses a somewhat different problem we do not discuss here. The work of Kakas and collaborators [5] uses the so-called "normal ILP learning settings" in which there is only one observation, (or database), and examples are ground predicates that have to be derived from a hypothesis, facts and background knowledge. To deal with uncertainty in I-ACL, some predicates are stated as "abducible", i.e. assumptions about their ground occurrences can be made in order to derive the examples. Note that, as there is one unique observation the assumptions concerning the examples are then mixed up, resulting in a complex combinatorial problem. This means that choices regarding sets of assumptions have to be made very early during learning. Note, however, that such abductive induction allows expressing dependencies between assumptions made on different examples.

6 Conclusion

The results of the experimentations show that LEAR copes well with high level of incompleteness. They also show that LEAR outperforms the standard closed-world approach as implemented in TILDE, at least when the available background knowledge induces a strong reduction on the number of ground clauses representing any ambiguous example. The program, written in swi-prolog, is efficient, and running times generally does not increase much with the ambiguity level. Still there is room to improve scalability.

References

[1] Hendrik Blockeel and Luc De Raedt. Top-down induction of first-order logical decision trees. *Artif. Intell.*, 101(1-2):285–297, 1998.

[2] Dominique Bouthinon, Henry Soldano, and Véronique Ventos. Concept learning from (very) ambiguous examples. In Petra Perner, editor, *MLDM*, volume 5632 of *Lecture Notes in Computer Science*, pages 465–478. Springer, 2009.

[3] Luc DeRaedt. Logical settings for concept-learning. *Artif. Intell.*, 95(1):187–201, 1997.

[4] Haym Hirsh. Generalizing version spaces. *Mach. Learn.*, 17(1):5–46, 1994.

[5] Antonis C. Kakas and Fabrizio Riguzzi. Abductive concept learning. *New Generation Computing*, 18(3):243–294, 2000.

[6] Alexandre Linhares. A glimpse at the metaphysics of bongard problems, 2000.

[7] S. Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13(3-4):245–286, 1995.

[8] Céline Rouveirol and Michèle Sebag, editors. *Inductive Logic Programming, 11th International Conference, ILP 2001, Strasbourg, France, September 9-11, 2001, Proceedings*, volume 2157 of *Lecture Notes in Computer Science*. Springer, 2001.

[9] Chiaki Sakama and Katsumi Inoue. Brave induction: a logical framework for learning from incomplete information. *Machine Learning*, 76(1):3–35, 2009.

[10] Jan Wielemaker. An overview of the SWI-Prolog programming environment. In Fred Mesnard and Alexander Serebenik, editors, *Proceedings of the 13th International Workshop on Logic Programming Environments*, pages 1–16, Heverlee, Belgium, december 2003. Katholieke Universiteit Leuven. CW 371.