Probabilistic Abductive Logic Programming using Possible Worlds

F. Rotella[1] and S. Ferilli[1,2]

[1] Dipartimento di Informatica – Università di Bari
{fulvio.rotella, stefano.ferilli}@uniba.it
[2] Centro Interdipartimentale per la Logica e sue Applicazioni – Università di Bari

Abstract Reasoning in very complex contexts often requires purely deductive reasoning to be supported by a variety of techniques that can cope with incomplete data. Abductive inference allows to guess information that has not been explicitly observed. Since there are many explanations for such guesses, there is the need for assigning a probability to each one. This work exploits logical abduction to produce multiple explanations consistent with a given background knowledge and defines a strategy to prioritize them using their chance of being true. Another novelty is the introduction of probabilistic integrity constraints rather than hard ones. Then we propose a strategy that learns model and parameters from data and exploits our Probabilistic Abductive Proof Procedure to classify never-seen instances. This approach has been tested on some standard datasets showing that it improves accuracy in presence of corruptions and missing data.

1 Introduction

In the field of *Artificial Intelligence* (AI) so far two approaches have been attempted: numerical/statistical on one hand, and relational on the other. Statistical methods are not able to fully seize the complex network of relationships, often hidden, between events, objects or combinations thereof. In order to apply AI techniques to learn/reason in the real world, one might be more interested on producing and handling complex representations of data than flat ones. This first challenge has been faced by exploiting First-Order Logic (FOL) for representing the world. This setting is useful when data are certain and complete. However this is far for being always true, there is the need for handling incompleteness and noise in data. Uncertainty in relational data makes things even more complex. In particular it can the affect the features, the type or more generally the identity of an object and the relationships in which it is involved. When putting together logical and statistical learning, the former provides the representation language and reasoning strategies, and the latter enforces robustness. This gave rise to a new research area known as *Probabilistic Inductive Logic Programming* [14] (PILP) or *Statistical Relational Learning* [6] (SRL). Although clearly relevant in complex domains such as Social or Biological data, it inherits well-known

problems from *Probabilistic Graphical Models* [11] (PGM) (parameter and model learning, inference).

Furthermore, reasoning in contexts characterized by a high degree of complexity often requires purely deductive reasoning to be supported by a variety of techniques that cope with the lack or incompleteness of the observations. Abductive inference can tackle incompleteness in the data by allowing to guess information that has not been explicitly observed [7]. For instance, if one is behind a corner and a ball comes out of it, a good explanation might be that someone has kicked it. However there are many other plausible explanations for the ball's movement, and thus that inference provides no certainty that someone really stroke the ball. While humans are able to discriminate which explanations are consistent with their previous knowledge and which ones have to be discarded, embedding in machines this capability is not easy.

This work faces two issues: the generation of multiple (and minimal) explanations consistent with a given background knowledge, and the definition of a strategy to prioritize different explanations using their chance of being true. It is organized as follows: the next section describes some related works; Section 3 introduces the Abductive Logic Programming framework; our Probabilistic Abductive Logic Proof procedure is presented in Section 4; then in Section 5 we propose a strategy to exploit our approach in classification tasks; finally there is an empirical evaluation on three standard datasets followed by some considerations and future works.

2 Related Work

Abductive reasoning is typically used to face uncertainty and incompleteness. In the literature there are two main approaches: uncertainty has been faced by bayesian probabilistic graphical models [13] and incompleteness by means of the classical approach based on pure logic [7]. Nowadays many works combined logical abduction and statistical inference, which allows to rank all possible explanations and choose the best one.

One of the earliest approaches is [12] where a program contains non-probabilistic definite clauses and probabilistic disjoint declarations $\{h_1 : p_1, ..., h_n : p_n\}$ where an abducible atom h_i is considered true with probability p_i and $i \in \{1, ..., N\}$. That work focuses on the representation language and proposes a simple language for integrating logic and probability. It does not integrate any form of logic-based abductive proof procedure with statistical learning, and considers hard assumptions for the nature of constraints (i.e. only disjoint declarations) in order to keep simple the general procedure. This framework does not assign probabilities to constraints but only to ground literals.

PRISM [16] is a system based on logic programming with multivalued random variables. It provides no support for integrity constraints but includes a variety of top-level predicates which can generate abductive explanations. Since a probability distribution over abducibles is introduced, the system chooses the

best explanation using a generalized Viterbi algorithm. Another central feature of PRISM is its capability to learn probabilities from training data.

Two approaches have merged directed and undirected graphical models with logic. The former [15] exploits Bayesian Logic Programs [14] (BLPs) as a representation language for abductive reasoning and uses the Expectation Maximization algorithm to learn the parameters associated to the model. The latter [9], exploiting Markov Logic Networks (MLN), carries out abduction by adding reverse implications for every rule in the knowledge base (since MLNs provide only deductive inference). However, the addition of these rules increases the size and complexity of the model, resulting computationally expensive. It's worth noting that like MLNs, most SRL formalisms use deduction for logical inference, and hence they cannot be used effectively for abductive reasoning.

An approach for probabilistic abductive Logic programming with Constraint Handling Rules has been proposed in [3]. It differs from other approaches to probabilistic logic programming by having both interaction with external constraint solvers and integrity constraints. Moreover exploits probabilities to optimize the search for explanations using Dijkstra's shortest path algorithm. Hence, the approach explores always the most probable direction, so that the investigation of less probable alternatives is suppressed or postponed. Although the knowledge representation formalism is similar to our one, there are several differences in the approaches. The first difference regards the support for negation. Their framework do not handle negation, and so they simulate it by introducing new predicate symbols (eg. haspower(X) $\rightarrow$ hasnopower(X)). The other difference involves the definition of the constraints that, due to the lack of negation, might be tricky. Conversely our ones allow a flexible and intuitive representation without such restrictions. The last difference is the lack of a strategy to learn the probabilities.

In [1] abduction is conducted with Stochastic Logic Programs [10] (SLP) by considering a number of *possible worlds*. Abductive reasoning is carried out by reversing the deductive flow of proof and collecting the probabilities associated to each clause. Although this approach is probabilistically consistent with the SLP language, abduction through reversion of deduction is quite hazardous because abductive reasoning by means of deduction without constraints may lead to wrong conclusions.

3 Abductive Logic Programming framework

Our proposal is based on Abductive Logic Programming [7] (ALP), a high-level knowledge representation framework that allows to solve problems declaratively based on abductive reasoning. It extends Logic Programming by allowing some predicates, called abducible predicates, to be incompletely defined. Problem solving is performed by deriving hypotheses on these abducible predicates (abductive hypotheses) as solutions of the problems to be solved. These problems can be either observations that need to be explained (as in classical abduction) or goals

to be achieved (as in standard logic programming). An abductive logic program is made up of a triple $< P, A, IC >$, where:

- P is a standard logic program;
- A (Abducibles) is a set of predicate names;
- IC (Integrity Constraints or domain-specific properties) is a set of first order formulae that must be satisfied by the abductive hypotheses.

These three components are used to define *abductive explanations.*

Definition 1 (Abductive explanation). *Given an abductive theory* $T =< P, A, IC >$ *and a formula* G*, an* abductive explanation Δ *for* G *is a set of ground atoms of predicates in* A *s.t.* $P \cup \Delta \models G$ *(*Δ *explains* G*) and* $P \cup \Delta \models IC$ *(*Δ *is consistent). When it exists,* T abductively entails G*, in symbols* $T \models_A G$*.*

Suppose a clause C: $h(t_1, .., t_n)$:- $l_1, ..l_{n'}$, h is the unique literal of the head, n is its arity, l_i with $i = \{1, ..., n'\}$ are the literals in the body and $\overline{l_i}$ stands for the negative literal $\neg l_i$. For instance, Example 1 defines a logic program P for the concept $printable(X)$ that describes the features that a document must own in order to be printed by a particular printer.

Example 1 (Example theory for paper domain).
$c_1 : printable(X) \leftarrow a4(X), \overline{text(X)}$
$c_2 : printable(X) \leftarrow a4(X), table(X), black\_white(X)$
$c_3 : printable(X) \leftarrow a4(X), text(X), black\_white(X)$
$A = \{image,\ text,\ black\_white,\ printable,\ table,\ a4,\ a5,\ a3\}$

In this framework, a proof procedure for abductive logic programs has been presented in [8]. It interleaves phases of *abductive* and *consistency derivations*: an *abductive derivation* is the standard Logic Programming derivation extended in order to consider abducibles. When an abducible literal δ has to be proved, it is added to the current set of assumptions (if not already included). Since the addition of δ must not violate any integrity constraint, a *consistency derivation* starts to check that all integrity constraints containing δ fail. In the *consistency derivation* an *abductive derivation* is used to solve each goal. This might cause an extension of the current set of assumptions.

More specifically an abductive derivation from $(G_1\ \Delta_1)$ to $(G_n\ \Delta_n)$ in $< P, A, IC >$ of a literal from a goal, is a sequence

$$(G_1\ \Delta_1), (G_2\ \Delta_2), ..., (G_n\ \Delta_n)$$

such that each G_i has the form $\leftarrow L_1, ..., L_k$ and $(G_{i+1}\Delta_{i+1})$ is obtained according to one of the following rules:

1. If L_j is not abducible, then $G_{i+1} = C$ and $\Delta_{i+1} = \Delta_i$ where C is the resolvent of some clause in P with G_i on the selected literal L_j;
2. If L_j is abducible and $L_j \in \Delta_i$, then $G_{i+1} = \leftarrow L_1, ..., L_{j-1}, L_{j+1}, ..., L_k$ and $\Delta_{i+1} = \Delta_i$;
3. If L_j is a ground abducible, $L_j \notin \Delta_i$ and $\overline{L_j} \notin \Delta_i$ and there exists a *consistency derivation* from $(\{L_j\}\ \Delta_i \cup \{L_j\})$ to$(\{\}\ \Delta')$ then $G_{i+1} = \leftarrow L_1, ..., L_{j-1}, L_{j+1}, ..., L_k$ and $\Delta_{i+1} = \Delta'$

In the first two steps the logical resolution is performed exploiting: (1) the rules of P and (2) the abductive assumptions already made. In the last step before adding a new assumption to the current set of assumption a consistency checking is performed.

A consistency derivation for an abducible α from (α, Δ_1) to $(F_n\ \Delta_n)$ in $< P, A, IC >$ is a sequence

$$(\alpha\ \Delta_1), (F_1\ \Delta_1), ..., (F_n\ \Delta_n)$$

where:

1. F_1 is the union of all goals of the form $\leftarrow L_1, ..., L_n$ obtained by resolving the abducible α with the constraints in IC with no such goal being empty;
2. for each $i > 1$ let F_i have the form $\{\leftarrow L_1, ..., L_k\} \cup F_i'$, then for some $j = 1, ..., k$ and each $(F_{i+1}\ \Delta_{i+1})$ is obtained according to one of the following rules:
   (a) If L_j is not abducible, then $F_{i+1} = C' \cup F_i'$ where C' is the set of all resolvents of clauses in P with $\leftarrow L_1, ..., L_k$ on literal L_j and the empty goal $[] \notin C'$, and $\Delta_{i+1} = \Delta$
   (b) If L_j is abducible, $L_j \in \Delta_i$ and $k > 1$, then $F_{i+1} = \{\leftarrow L_1, ..., L_k\} \cup F_i'$ and $\Delta_{i+1} = \Delta_i$
   (c) If L_j is abducible, $\overline{L_j} \in \Delta_i$ then $F_{i+1} = F_i'$ and $\Delta_{i+1} = \Delta_i$
   (d) If L_j is a ground abducible, $L_j \notin \Delta_i$ and $\overline{L_j} \notin \Delta_i$ and there exists an *abductive derivation* from $(\leftarrow \overline{L_j}\ \Delta_i)$ to $([]\ \Delta')$ then $F_{i+1} = F_i'$ and $\Delta_{i+1} = \Delta'$;
   (e) If L_j is equal to $\overline{A}$ with A a ground atom and there exists an *abductive derivation* from $(\leftarrow \overline{A}\ \Delta_i)$ to $([]\ \Delta')$ then $F_{i+1} = F_i'$ and $\Delta_{i+1} = \Delta'$.

In the first case 2a the current branch is split into the number of resolvents of $\leftarrow L_1, ..., L_k$ with the clauses in P on L_j. If we get the empty clause the whole check fails. In the second case if L_j belongs to Δ_i, it is discarded and if it is alone, the derivation fails. In case 2c the current branch is consistent with the assumptions in Δ_i and so it is dropped from the checking. In the last two cases 2d and 2e the current branch can be dropped if respectively $\leftarrow \overline{L_j}$ and A are abductively probable.

The procedure returns the minimal abductive explanation set if any, otherwise it fails. It is worth noting that according to the implementation in [8] the Δs in the abductive explanations must be ground. Thus if a non ground abducible is encountered it is first unified with a clause in the background knowledge, with an example or with a previously abduced literal. If this is not possible, it is grounded with a skolem constant.

4 Probabilistic Abductive Logic Programming

This work extends the technique shown in Section 3 in order to smooth the classical rigid approach with a statistical one.

In order to motivate our approach we can assume that to abduce a fact, there is the need for checking if there are constraints that prevent such an abduction.

The constraints can be either universally valid laws (such as temporal or physical ones), or domain-specific restrictions. Constraints verification can involve other hypotheses that have to be abduced, and others that can be deductively proved. We can be sure that if a hypothesis is deductively verified, it can be surely assumed true. Conversely, if a hypothesis involves other abductions, there is the need of evaluating all possible situations before assuming the best one. In this view each abductive explanation can be seen as a possible world, since each time one assumes something he conjectures the existence of that situation in a specific world. Some abductions might be very unlikely in some worlds, but most likely in other ones. The likelihood of an abduction can be assessed considering what we have seen in the real world and what we should expect to see.

Moreover, this work handles a new kind of constraints since typically the constraints are only the nand of the conditions and are not probabilistic. They allow a more suitable and understandable combination of situations. The first kind is the classical *nand* denial where at least one condition must be false, the type *or* represent a set of conditions where at least one must be true, and the type *xor* requires that only one condition must be true. Moreover, due to noise and uncertainty in the real world, we have smoothed each constraint with a probability that reflects the degree of the personal belief in the likelihood of the whole constraint.

In Example 2 it can be seen that each probabilistic constraint is a triple $\langle Pr, S, T\rangle$ where Pr is a probability and expresses its reliability, or our confidence on it, T is the *type* of constraint and represents the kind of denial, and S is the set of literals of the constraint. For example ic_1 states that a document can be only of one of the three size formats (a3, a4 or a5) and that our personal belief in the likelihood of this constraint is 0.8, ic_2 states that a document can be composed either of tables, images or text, and so on.

Example 2 (Typed Probabilistic Constraints).
$ic_1 = \langle 0.8, [a3(X), a4(X), a5(X)], xor\rangle$
$ic_2 = \langle 0.9, [table(X), text(X), image(X)], or\rangle$
$ic_3 = \langle 0.3, [text(X), color(X)], nand\rangle$
$ic_4 = \langle 0.3, [table(X), color(X)], nand\rangle$
$ic_5 = \langle 0.6, [black\_white(X), color(X)], xor\rangle$

Our probabilistic approach to logical abductive reasoning can be described from two perspectives: the logical proof procedure and the computation of probabilities.

4.1 ALP perspective

The logical proof procedure consists of an extended version of the classical one [7, 8]. While the classical procedure resolved the goal by looking for one minimal abductive explanation set, our approach is to generate different (minimal) abductive explanations and then evaluate them all. This goal can be achieved by changing each assumption that may constrain the subsequent abductive explanations. For example, supposing that a document is black and white, all subsequent

assumptions must be consistent with this. But if we change this assumption, assuming that in another *world* that document is not black and white, we might obtain another set of hypoteses consistent with this last assumption. Each time the procedure assumes something two *possible worlds* can be considered: one where the assumption holds and another where it does not. The overall view is analogous to the exploration of a tree in which each interior node corresponds to a decision (an assumption), as for example changing the truth value of a literal, an edge represents the consequences of that decision and a leaf is the conclusion of the abductive reasoning in a particular *possibile consistent world*. In order to generate such *possible worlds* in which a literal holds and others in which it does not, the classical procedure has been extended by means of introducing two rules for the derivation procedures. The 4th rule of the *abductive derivation* will be:

4. If $\overline{L_j}$ is a ground abducible, $\overline{L_j} \notin \Delta_i$ and $L_j \notin \Delta_i$ and there exists a *consistency derivation* from $(\{\overline{L_j}\}\ \Delta_i \cup \{\overline{L_j}\})$ to $(\{\}\ \Delta')$ then $G_{i+1} = \leftarrow L_1, ..., L_{j-1}, L_{j+1}, ..., L_k$ and $\Delta_{i+1} = \Delta'$

A corresponding rule 2d* for the *consistency derivation* is introduced between 2d and 2e, and states:

2d* If $\overline{L_j}$ is a ground abducible, $L_j \notin \Delta_i$ and $\overline{L_j} \notin \Delta_i$ and there exists an *abductive derivation* from $(\leftarrow L_j\ \Delta_i)$ to $([]\ \Delta')$ then $F_{i+1} = F'_i$ and $\Delta_{i+1} = \Delta'$;

It can be noted that the rules 3 and 4 of the *abductive derivation* are the choice points on which the procedure can backtrack in order to assume both values (positive and negative) of a literal. In fact in a *possible world* L_j is added to the set of hypoteses by means of 3rd rule, and in the other *possible world* $\overline{L_j}$ holds by means of 4th rule. The analogous choice points in the *consistency derivation* are respectively the rules 2d* and 2d.

The choice of the predicate definition, that is the 1st rule of the *abductive derivation*, is another choice point where the procedure can backtrack to explore different explanations and consequently other *possible worlds*. In fact choosing different definitions, other literals will be taken into account and thus other constraints must be satisfied, and so on. It is worth noting that if some assumptions do not preserve the consistency, the procedure discards them along with the corresponding *possible worlds*. Section 4.2 will present a probabilistic strategy to choose the most likely explanation among all *possible worlds*.

Example 3 (Observation o_1, Query and Possible Explanations).

$a4(o_1)$	?- $printable(o_1)$
$\Delta_1 = \{\overline{text(o_1)}, table(o_1)\}$	$Ic_1 = \{ic_2, ic_3, ic_4\}$
$\Delta_2 = \{text(o_1), \overline{table(o_1)}, image(o_1)\}$	$Ic_2 = \{ic_2, ic_3, ic_4\}$
$\Delta_3 = \{table(o_1), black\_white(o_1)\}$	$Ic_3 = \{ic_2, ic_4, ic_5\}$
$\Delta_4 = \{text(o_1), black\_white(o_1)\}$	$Ic_4 = \{ic_2, ic_3, ic_5\}$

In order to motivate our point of view, let's consider Example 3 belonging to the "paper" domain presented in Section 3. In order to abductively cover o_1 by means of the concept definition $printable(X)$ (i.e. query ?- $printable(o_1)$), there is the need for abducing other literals since the concept definitions c_i with $i = \{1, ..., 3\}$ need some facts that are not in the knowledge base (KB) (i.e. only

$a4(o_1)$ holds). This is the classical situation in which deductive reasoning fails and there is the need of abductive one.

The procedure executes an *abductive derivation* applying the 1st rule and obtaining one of the resolvent in P, for instance, the first clause $c_1 : printable(X) \leftarrow a4(X), \overline{text(X)}$. In this case $a4(o_1)$ holds, and so we can exclude it from the abductive procedure continuing with the next literal $\overline{text(o_1)}$. This literal does not hold, and so there is the need of abducing it. This abduction involves the verification of ic_2 and ic_3 by the *consistency derivation*. Since ic_2 is a *or* constraint and thus at least one literal must be true, there are two possible ways to satisfy it considering that $text(o_1)$ must not hold (by current hypothesis). Thus there are two *possible worlds*, one in which $table(o_1)$ holds, and the other in which $table(o_1)$ does not.

In the former *world* rule 2d* fires and $table(o_1)$ is abduced by performing a *consistency derivation* on the constraints ic_2 and ic_4. The former constraint is already verified by means of the previous abductions (rule 2b). Since ic_4 is a *nand* constraint and thus at least one literal must be false, it is satisfied because *color* is not abducible (see abducible predicates A in Example 1) and does not hold (if a literal is not abducible, its value depends on background knowledge, i.e. rule 2a). Thus coming back to the initial abduction $\overline{text(o_1)}$, ic_2 is satisfied abducing $table(o_1)$ and ic_3 is already satisfied (since it is a *nand* constraint, it is falsified by abducing $\overline{text(o_1)}$). The first abductive explanation Δ_1 for the goal $printable(o_1)$ is so formed by the abductions of $\overline{text(o_1)}$ and $table(o_1)$ (see Example 3).

Similarly to the classical procedure that in backtracking returns all minimal abductive explanations, our procedure comes back to each choice point changing the truth values of the literals in order to explore different possible explanations (and thus other possible worlds). The first choice point, as mentioned above, regards the abduction of $table(o_1)$. While in the former *possible world* $table(o_1)$ holds, now the procedure abduces $\overline{table(o_1)}$ (rule 2d) thus exploring the other *possible worlds*. In this *world* to abduce $\overline{table(o_1)}$ the constraints ic_2 and ic_4 are taken into account. The former constraint is verified by abducing $\overline{image(o_1)}$ (i.e. *or* constraint) and the latter is satisfied by the initial abduction $\overline{text(o_1)}$ as in explanation Δ_1. Then $\overline{text(o_1)}$ can be abduced also in this *world* since ic_2 has been just verified and ic_3 as before. So Δ_2 is the second explanation obtained by changing the assumption on literal $table(o_1)$ and thus exploiting another *possible world*.

The explanations Δ_1 and Δ_2 are the only two *possible worlds* given the first clause c_1 because other literal configurations are inconsistent. Hence, the procedure comes back to last backtracking point that were the choice of the predicate definition. This time the 1st rule of the *abductive derivation* can be applied to obtain the second predicate definition $c_2 : printable(X) \leftarrow a4(X), table(X), black\_white(X)$. So there is the need of abducing $table(o_1)$ and $black\_white(o_1)$. The former can be abduced because ic_2 is satisfied (i.e. *or* constraint), and ic_4 is verified by means of $color(o_1)$. The latter literal can be abduced because ic_5 is a *xor* constraint and thus at most one literal must be true. In explanation Δ_3 no other

literals must be abduced by the *consistency derivation* and so no other worlds must be explored since the abduced literals $table(o_1)$ and $black\_white(o_1)$ are constrained by the predicate definition.

Once again the procedure comes back to last backtracking point and chooses the predicate definition $c_3 : printable(X) \leftarrow a4(X), text(X), black\_white(X)$. The *abductive derivation*, similarly to the previous run, returns only one possible explanation Δ_4 as it can been seen in Example 3.

It easy to note that rules (2d) and (2d*) are candidate entry points for backtracking in the *consistency derivation* , and rules 1,3 and 4 are the analogues in the *abductive derivation*. In this way all possible (minimal) explanations are obtained along with all *possible consistent worlds*.

4.2 Probabilistic perspective

After all different explanations have been found by the above abductive proof procedure, the issue of selecting the best one arises. The simple approach of choosing the minimal explanation is reliable when there is only one minimal proof or when there are no ways to assess the reliability of the explanations. However, Example 3 shows that there might be different explanations for the same observation and so we need to assess the probability of each explanation in order to choose the best one. To face this issue, our approach regards each set of abductions as a *possible world* and so a chance of being true can be assigned to it. The abduction probability of each ground literal through *the possible worlds* can be obtained considering two aspects: the chance of being true in the real world and all sets of assumptions made during the *consistency derivation* in each *possible world*.

Let's introduce some notation: $\Delta = \{P_1 : (\Delta_1, Ic_1), ..., P_T : (\Delta_T, Ic_T)\}$ is the set of the T consistent *possible worlds* that can be assumed for proving a goal G (i.e. the observation to be proved). Each (Δ_i, Ic_i) is a pair of sets: $\Delta_i = \{\delta_1, ..., \delta_J\}$ contains the ground literals δ_j with $j \in \{1, ..., J\}$ abduced in a single abductive proof, and $Ic_i = \{ic_1, ..., ic_K\}$ is the set of the constraints ic_k with $k = \{1, ..., K\}$ involved in the explanation Δ_i. Both Δ_i and Ic_i may be empty. Moreover, we have used the following symbols in our equations: $n(\delta_j)$ is the number of true grounding of the predicate used in literal δ_j, $n(cons)$ is total number of constants encountered in the world, $a(\delta_j)$ is the arity of literal δ_j and $P(ic_k)$ is the probability of the kth-constraint. Thus the chance of being true of a ground literal δ_j can be defined as:

$$P(\delta_j) = \frac{n(\delta_j)}{\frac{n(cons)!}{(n(cons)-a(\delta_j))!}} \tag{1}$$

Then the unnormalized probability of the abductive explanation can be assessed by Equation 2 and the probability of the abductive explanation normalized over

all T consistent worlds can be computed as in Equation 3:

$$P'_{(\Delta_i, Ic_i)} = \prod_{j=1}^{J} P(\delta_j) * \prod_{k=1}^{K} P(ic_k) \quad (2) \qquad P_{(\Delta_i, Ic_i)} = \frac{P'_{(\Delta_i, Ic_i)}}{\sum_{t=1}^{T} P'_{(\Delta_t, Ic_t)}} \quad (3)$$

Equation 1 expresses the ratio between true and possible groundings of literal δ_j. The intuition behind this equation can be expressed with a simple example: given a feature $f(\cdot)$ and an item *obj* that does not have such a feature, if we want to assess the probability that *obj* owns $f(\cdot)$ (i.e. $P(f(obj))$), we should consider how often we found items that hold $f(\cdot)$ over all items that might own it in real world.

It is worth noting that we are considering the Object Identity [17] assumption and so within a clause, terms (even variables) denoted with different symbols must be distinct (i.e. they must refer to different objects). Thus only literal groundings without constants repetitions are allowed in Equation 1. It is important to underline that such a bias does not limit the expressive power of the language, since for any clause/program it is possible to find an equivalent version under Object Identity. The first part of the formula 2 encloses the probability that all abduced literals are true in that particular *world*. The second part expresses the reliability of the constraints involved in the i-th abductive explanation. Although our approach is focused on the computation of the *most probable explanation* and hence there is no need of normalizing the probabilities of the explanations over all possible worlds (i.e some worlds are ruled out due to the presence of integrity constraints), it follows that Equation 3 is presented for completeness. The probability of $\overline{\delta_j}$ is equal to $1 - P(\delta_j)$.

Example 4 (Probability assessment of the Abductive Explanations).
$A = \{0.2{:}image, 0.4{:}text, 0.1{:}black\_white, 0.6{:}printable, 0.1{:}table, 0.9{:}a4, 0.1{:}a5, 0.1{:}a3\}$
$P'(\Delta_1, Ic_1) = 0.00486 \quad P'(\Delta_2, Ic_2) = 0.00875$
$P'(\Delta_3, Ic_3) = 0.00162 \quad P'(\Delta_4, Ic_4) = 0.00648$

For the sake of clarity, each abducible literal of the current example has been labeled with its probability using formula (1) as shown in Example 4. For instance, the probability of the first explanation using (2) can be computed as $P'_{(\Delta_1, Ic_1)} = P(\overline{text(o_1)}) * P(table(o_1)) * P(ic_2) * P(ic_3) * P(ic_4)$ and thus $P'_{(\Delta_1, Ic_1)} = 0.6 * 0.1 * 0.9 * 0.3 * 0.3 = 0.00486$. Then, Example 4 shows the probability of all explanations computed using equation (2).

Finally we can state the maximum probability among all abductive explanations. It corresponds to the maximum between all T possible consistent worlds (in this example T is equal to 4) s.t. $P'(printable(o_1)) = \max_{1 \leq i \leq T} P'_i(\Delta_i, Ic_i)$, that is Δ_2. It is worth noting that this behaviour claims the need of the logical abductive reasoning to be supported by a probabilistic assessment of all abductive explanations rather than relying on the minimal one.

5 Improving Classification Exploiting Probabilistic Abductive Reasoning

Now the above proof procedure can be exploited to classify never-seen instances. In particular we first learns from data the model (i.e. the Abductive Logic Program $< P, A, IC >$) and the parameters (i.e. literals probabilities), and then our Probabilistic Abductive Logic proof procedure can be exploited to classify new instances. The strategy, presented in Algorithm 1, can be split into two

Algorithm 1 Probabilistic Classification Algorithm

Require: A is the set of abducibles, a couple $<Train_i, Test_i>$
Ensure: $Pred_i$, the set of examples labelled with most likely class.

```
1: T_i ← learn_background_theory(Train_i)
2: IC_i ← learn_integrity_constraints(Train_i)
3: ProbLit_i ← compute_literals_probabilities(Test_i)
4: Pred_i = ∅
5: for each example e in Test_i do
6:    R = ∅
7:    for each class c in T_i do
8:       <P(c,e), Δ_p> ← probabilistic_abductive_proof(ProbLit_i, c, e)
9:       <P(¬c,e), Δ_n> ← probabilistic_abductive_proof(ProbLit_i, ¬c, e)
10:      if P(c,e) > P(¬c,e) then
11:         R ← R ∪ <P(c,e), Δ_p>
12:      else if P(c,e) < P(¬c,e) then
13:         R ← R ∪ <P(¬c,e), Δ_n>
14:      else
15:         discard e, inconsistency
16:   <P(c*,e), Δ*> ← most_likely_class_in(R)
17:   Pred_i ← Pred_i ∪ <P(c*,e), Δ*>
```

parts: the former prepares the data (model and parameters), and the latter performs the classification. In the former part, given a train set $Train_i$ and a set of abducible literals A (possibly empty) our approach learns the corresponding Theory T_i (line 1) by exploiting INTHELEX [4], a well-known ILP system, and then obtains the integrity constraints (line 2) with the procedure described in [5]. Such procedure returns a set of *nand* constraints of different sizes and descriptor type domain. This last information can be useful to define a new kind of constraint called *type_domain* that can be dealt as an *xor* constraint. In fact if the descriptor type domain for the color property is {blue, red, yellow, black, green}, and the object X is part of an observation, it will be impossible to abduce two different color descriptors from the above set applied to X. However since our procedure handles natively a probability value associated to the constraints, and this procedure does not return those values, the constraints should be manually labelled or they will be automatically considered true with probability of 1.0. In any case, the set of abducible A can be left empty, because our system considers

abducible all predicates without definitions in T_i. The last step of the first part (line 3) computes the Equation 1 for each literal in $Train_i$ before starting the abductive procedure since those values depend only on $Train_i$.

Hence, the second part of the strategy starts at line 5. As can be seen at lines 8 and 9, our algorithm tries to abductively cover the example considering both as positive and as negative for the class c. In fact, when an example is considered negative, our procedure discovers all *possibile worlds* in which it cannot be abductively proved as instance of concept c. Specularly if the example is positive, it discovers all the *possibile worlds* in which must be abduced something to prove it. Those two executions return an explanation probability that can be compared each other in order to choose the best class. Then the algorithm selects the best classification between all concept probabilities as can be seen at line 16.

It is worth noting that this strategy cannot be performed by a pure abductive logic proof procedure since in such context we do not need a logical response (true/false) but we want a probabilistic value.

6 Experimental Evaluation

The evaluation is aimed at assessing the quality of the results obtained by the probabilistic classification when it faces incomplete and noisy data. All experiments were performed on datasets obtained from UCI [2] machine learning repository.

A 10-fold split of each dataset has been performed in order to obtain a set of 10 couples $<Train, Test>$. Then each test-set has been replaced by a set of corrupted versions, in which we removed at random a K% of each example description, with K varying from 10% to 70% with step 10. This procedure has been repeated 5 times for each corrupted test-set in order to randomize the example corruption. In this way 35 test-sets for each fold have been obtained (7 levels of corruption by 5 runs for level). In order to compare the outcomes with a complete test-set we exploited deductive reasoning on the original test-set (i.e corruption level 0%). Moreover we exploited only deductive reasoning to the same corrupted test-set in order to show the improvement of our approach. The maximum length of constraints has been set to 4 for all datasets. Since we do not have any previous knowledge on the datasets we assumed true all obtained constraints with a probability of 1.0 as described in the Section 5.

Then the performances of the system can be evaluated with the aim of understanding how the approach is sensible to the progressive lack of knowledge across the 10 folds. The following synthetic descriptions of the datasets refer to values averaged on the folds.

Breast-Cancer contains 201 instances of the benign class and 85 instances of the malignant class and each instance is described by 9 literals. There is the presence of less than 10 instances with missing values. The learned theory consists of 30 clauses where each one has an average of 6 literals in the body. The learned integrity constraints are 1784 (55% are constraints of length 4, 35%

of length 3 and 10% of length 2), and 9 type domain constraints (one for each literal in the example description language).

Congressional Voting Records contains 435 instances (267 democrats, 168 republicans) classified as democrats or republicans according to their votes. Each instance is described by 16 literals. The obtained theory consists of 35 clauses where each one has an average of 4.5 literals in the body. The learned integrity constraints are 4173 (16% are constraints of length 4, 37% of length 3 and 47% of length 2), and 16 type domain constraints (as before).

Tic-Tac-Toe dataset contains 958 end game board configurations of tic-tac-toe (about 65.3% are positive), where x is assumed to have played first. The target concept win(x) represents one of the 8 possible ways to create a three-in-a-row. Thus, each instance is described by 8 literals. The learned theory consists of 18 clauses where each has an average of 4 literals in the body. The learned integrity constraints are 1863 (99% are constraints of length 4, 1% of length 3), and 16 type domain constraints (as before).

Results and Discussion

Figure 1 shows the average accuracy obtained on each dataset with respect to

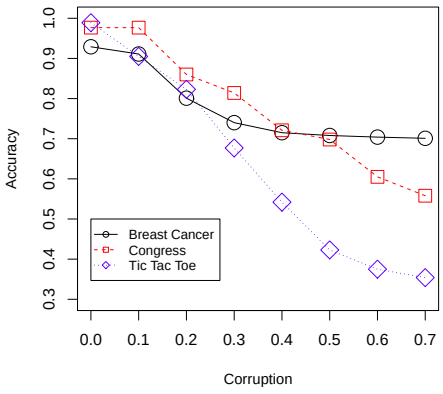

Figure 1: Average Accuracy Curves

Dataset	Corr.	Abductive Reas.			Deductive Reas.		
		Prec.	*Rec.*	F_1	*Prec.*	*Rec.*	F_1
Breast	0%	0.891	0.870	0.881	0.891	0.870	0.881
	10%	0.865	0.835	0.850	0.634	0.454	0.227
	20%	0.853	0.411	0.556	0.571	0.118	0.195
	30%	0.800	0.188	0.584	0.500	0.029	0.056
	40%	1.000	0.059	0.111	—	—	—
	50%	1.000	0.035	0.068	—	—	—
	60%	1.000	0.023	0.046	—	—	—
	70%	1.000	0.012	0.023	—	—	—
Congress	0%	1.000	0.961	0.980	1.000	0.961	0.980
	10%	1.000	0.961	0.981	0.971	0.793	0.873
	20%	1.000	0.769	0.869	0.971	0.761	0.853
	30%	1.000	0.680	0.809	0.982	0.714	0.827
	40%	1.000	0.538	0.700	0.979	0.623	0.761
	50%	1.000	0.500	0.667	1.000	0.425	0.596
	60%	1.000	0.346	0.514	1.000	0.333	0.500
	70%	1.000	0.269	0.424	1.000	0.264	0.418
TikTakToe	0%	1.000	0.983	0.992	1.000	0.983	0.992
	10%	1.000	0.833	0.909	0.842	0.743	0.789
	20%	1.000	0.730	0.844	0.808	0.531	0.641
	30%	1.000	0.508	0.673	0.796	0.387	0.521
	40%	1.000	0.302	0.463	0.829	0.261	0.397
	50%	1.000	0.127	0.225	0.697	0.103	0.180
	60%	1.000	0.048	0.090	0.777	0.031	0.060
	70%	1.000	0.016	0.031	1.000	0.004	0.009

Table 1: Results of the experiments

the corruption levels. The probabilistic classification performed on the first two datasets allows to assess the strength and robustness of the approach. In fact their accuracy curves go down less fast than the third's one and even when the

70% of each example description has been removed, the system is able to classify never-seen instances with a mean accuracy of 0.7 and 0.6 respectively.

In order to understand the non-linear trend of the accuracy on the third dataset we can analyze Table 1. It shows the classification performances averaged within the same fold (i.e. 5 random corruptions) and between different folds for each level of corruption. We can find a sharp decay of recall for the third dataset in correspondence of the descending accuracy curve. This happens for two reasons: each example is less described than the ones of the other datasets (8 literals for description) and its background theory consists of an average of 4 literals for clause. Those two aspects make the approach more prone to abduce few literals (often a single one) to make positive examples never covered by the class definitions.

Comparing these results with the deductive approach, it can be noted that in the first dataset after the 30% of corruption, no positive example has been classified correctly. This deficiency has never affected our approach in all experiments. The performances of the deductive approach on last two datasets degrade faster compared to the abductive one.

In general if the examples are less described (as in Tic-Tac-Toe and Breast-Cancer), the number of misclassifications increases due to missing information. It follows that corruption levels greater than 50% rise up the chance of removing important object features, and so such experiments have been performed only for putting stress on the proposed approach.

7 Conclusions

Reasoning in very complex contexts often requires pure deductive reasoning to be supported by a variety of techniques that can cope with incomplete and uncertain data. Abductive inference allows to guess information that has not been explicitly observed. Since there are many explanations for such guesses, there is the need for assigning a probability to them in order to choose the best one.

In this paper we propose a strategy to rank all possible minimal explanations according to their chance of being true. We have faced two issues: the generation of multiple explanations consistent with the background knowledge, and the definition of a strategy to prioritize among different ones using their chance of being true according to the notion of *possible worlds*. Another novelty has been the introduction of probabilistic integrity constraints rather than hard ones as in the classical Abductive Logic Programming framework.

The proposal has been described from two perspectives: the abductive proof procedure and the computation of the probabilities. The former, extending the classical one, allows to generate many different explanations for each abduction, while the latter provides a probabilistic assessment of each explanation in order to choose the most likely one. Moreover we introduce a strategy to exploit our Probabilistic Abductive Proof procedure in classification tasks. The approach has been evaluated on three standard datasets, showing that it is able to cor-

rectly classify unseen instances in the presence of noisy and missing information. Further studies will be focused on learning the probabilistic constraints, and on the use of a richer probabilistic model for the literal probability distribution. Then we aim to exploit the Probabilistic Abductive Proof Procedure in other tasks such as natural language understanding and plan recognition.

References

[1] Andreas Arvanitis, Stephen H. Muggleton, Jianzhong Chen, and Hiroaki Watanabe. Abduction with stochastic logic programs based on a possible worlds semantics. In *In Short Paper Proc. of 16th ILP*, 2006.

[2] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[3] Henning Christiansen. Implementing probabilistic abductive logic programming with Constraint Handling Rules. pages 85–118. 2008.

[4] Floriana Esposito, Giovanni Semeraro, Nicola Fanizzi, and Stefano Ferilli. Multistrategy theory revision: Induction and abduction in inthelex. *Machine Learning*, 38:133–156, 2000.

[5] Stefano Ferilli, Teresa M. A. Basile, Nicola Di Mauro, and Floriana Esposito. Automatic induction of abduction and abstraction theories from observations. In *Proc. of the 15th ILP*, ILP'05, pages 103–120, Berlin, Heidelberg, 2005. Springer-Verlag.

[6] Lise Carol Getoor. *Learning statistical models from relational data*. PhD thesis, Stanford, CA, USA, 2002. AAI3038093.

[7] A. C. Kakas, R. A. Kowalski, and F. Toni. Abductive logic programming, 1993.

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

[9] Rohit J. Kate and Raymond J. Mooney. Probabilistic abduction using markov logic networks. In *Proceedings of the IJCAI-09 Workshop on Plan, Activity, and Intent Recognition (PAIR-09)*, Pasadena, CA, July 2009.

[10] S. Muggleton. Stochastic Logic Programs. In L. De Raedt, editor, *Proceedings of the 5th International Workshop on Inductive Logi c Programming*. Department of Computer Science, Katholieke Universiteit Leuven, 1995.

[11] Judea Pearl. Graphical models for probabilistic and causal reasoning. In *The Computer Science and Engineering Handbook*, pages 697–714. 1997.

[12] David Poole. Probabilistic horn abduction and bayesian networks. *Artif. Intell.*, 64(1):81–129, 1993.

[13] David Poole. Learning, Bayesian probability, graphical models, and abduction. In *Abduction and Induction: Essays on their Relation and Integration, Chapter 10*, pages 153–168. Kluwer, 1998.

[14] Luc De Raedt and Kristian Kersting. Probabilistic inductive logic programming. In *ALT*, pages 19–36, 2004.

[15] Sindhu V. Raghavan. Bayesian abductive logic programs: A probabilistic logic for abductive reasoning. In Toby Walsh, editor, *IJCAI*, pages 2840–2841, 2011.

[16] Taisuke Sato. Em learning for symbolic-statistical models in statistical abduction. In *Progress in Discovery Science, Final Report of the Japanese Discovery Science Project*, pages 189–200, London, UK, UK, 2002. Springer-Verlag.

[17] Giovanni Semeraro, Floriana Esposito, Donato Malerba, Nicola Fanizzi, and Stefano Ferilli. A logic framework for the incremental inductive synthesis of datalog theories. In Norbert E. Fuchs, editor, *LOPSTR*, volume 1463 of *Lecture Notes in Computer Science*, pages 300–321. Springer, 1997.