Learning Possibilistic Logic Theories from Default Rules

Ondřej Kuželka
Cardiff University, UK
KuzelkaO@cardiff.ac.uk

Jesse Davis
KU Leuven, Belgium
jesse.davis@cs.kuleuven.be

Steven Schockaert
Cardiff University, UK
SchockaertS1@cardiff.ac.uk

Abstract

We introduce a setting for learning possibilistic logic theories from defaults of the form “if alpha then typically beta”. We first analyse this problem from the point of view of machine learning theory, determining the VC dimension of possibilistic stratifications as well as the complexity of the associated learning problems, after which we present a heuristic learning algorithm that can easily scale to thousands of defaults. An important property of our approach is that it is inherently able to handle noisy and conflicting sets of defaults. Among others, this allows us to learn possibilistic logic theories from crowdsourced data and to approximate propositional Markov logic networks using heuristic MAP solvers. We present experimental results that demonstrate the effectiveness of this approach.

1 Introduction

Structured information plays an increasingly important role in applications such as information extraction [Dong et al., 2014], question answering [Kalyanpur et al., 2012] and robotics [Beetz et al., 2011]. With the notable exceptions of CYC and WordNet, most of the knowledge bases that are used in such applications have at least partially been obtained using some form of crowdsourcing (e.g. Freebase, Wikidata, ConceptNet). To date, such knowledge bases are mostly limited to facts (e.g. Obama is the current president of the US) and simple taxonomic relationships (e.g. every president is a human). One of the main barriers to crowdsourcing more complex domain theories is that most users are not trained in logic. This is exacerbated by the fact that often (commonsense) domain knowledge is easiest to formalize as defaults (e.g. birds typically fly), and, even for non-monotonic reasoning (NMR) experts, it can be challenging to formulate sets of default rules without introducing inconsistencies (w.r.t. a given NMR semantics) or unintended consequences.

In this paper, we propose a method for learning consistent domain theories from crowdsourced examples of defaults and non-defaults. Since these examples are provided by different users, who may only have an intuitive understanding of the semantics of defaults, together they will typically be inconsistent. The problem we consider is to construct a set of defaults which is consistent w.r.t. the System P semantics [Kraus et al., 1990], and which entails as many of the given defaults and as few of the non-defaults as possible. Taking advantage of the relation between System P and possibilistic logic [Benferhat et al., 1997], we treat this as a learning problem, in which we need to select and stratify a set of propositional formulas.

The contributions of this paper are as follows. First, we show that the problem of deciding whether a possibilistic logic theory exists that perfectly covers all positive and negative examples is $\Sigma_2^p$-complete. Second, we formally study the problem of learning from defaults in a standard learning theory setting and we determine the corresponding VC-dimension, which allows us to derive theoretical bounds on how much training data we need, on average, to obtain a system that can classify defaults as being valid or invalid with a given accuracy level. Third, we introduce a heuristic algorithm for learning possibilistic logic theories from defaults and non-defaults. To the best of our knowledge, our method is the first that can learn a consistent logical theory from a set of noisy defaults. We evaluate the performance of this algorithm in two crowdsourcing experiments. In addition, we show how it can be used for approximating maximum a posteriori (MAP) inference in propositional Markov logic networks [Richardson and Domingos, 2006; Dupin de Saint-Cyr et al., 1994]. An online appendix to this paper with additional details is available.\footnote{http://arxiv.org/abs/1604.05273}

2 Related work

Reasoning with defaults of the form “if alpha then typically beta”, denoted as $\alpha \sim \beta$, has been widely studied [Kraus et al., 1990; Pearl, 1990; Lehmann and Magidor, 1992; Geffner and Pearl, 1992; Goldszmidt et al., 1993; Benferhat et al., 1997]. A central problem in this context is to determine what other defaults can be derived from a given input set. Note, however, that the existing approaches for reasoning about default rules all require some form of consistency (e.g. the input set cannot contain both $a \sim b$ and $a \sim \neg b$). As a result, these approaches cannot directly be used for reasoning about noisy crowdsourced defaults.

To the best of our knowledge, this is the first paper that considers a machine learning setting where the input consists of default rules. Several authors have proposed ap-
proaches for constructing possibility distributions from data; see [Dubois and Prade, 2015] for a recent survey. How-
however, such methods are generally not practical for constructing possibilistic logic theories. The possibilistic counter-
part of the Z-ranking constructs a possibilistic logic theory from a set of defaults, but it requires that these defaults
are consistent and cannot handle non-defaults [Benferhat et al., 1997], although an extension of the Z-ranking that can
cope with non-defaults was proposed in [Booth and Paris, 1998]. Some authors have also looked at the problem of
learning sets of defaults from data [Benferhat et al., 2003; Kern-Isberner et al., 2008], but the performance of these
methods has not been experimentally tested. In [Serrurier and Prade, 2007], a possibilistic inductive logic programming
(ILP) system is proposed, which uses a variant of possibilistic logic for learning rules with exceptions. However, as is
common for ILP systems, this method only considers classification problems, and cannot readily be applied to learn gen-
eral possibilistic logic theories. Our work can also be seen as conceptually related to belief merging, although we are not
aware of any existing methods for merging inconsistent sets of default rules. Finally note that the setting of learning from
default rules as introduced in this paper can be seen as a non-
monotonic counterpart of an ILP setting called learning from entailment [De Raedt, 1997].

3 Background

3.1 Possibilistic logic

A stratification of a propositional theory \( T \) is an ordered par-
tition of the set of formulas in \( T \). A theory in possibilis-
tic logic [Dubois et al., 1994] is a set of formulas of the form \((\alpha, \lambda)\), with \( \alpha \) a propositional formula and \( \lambda \in [0,1] \)
a certainty weight. These certainty weights are interpreted in a purely ordinal fashion, hence a possibilistic logic the-
ory is essentially a stratification of a propositional theory. The strict \( \lambda \)-cut \( \Theta_\lambda \) of a possibilistic logic theory \( \Theta \) is de-
fined as \( \Theta_\lambda = \{ \alpha | (\alpha, \mu) \in \Theta, \mu > \lambda \} \). The inconsis-
tency level \( inc(\Theta) \) of \( \Theta \) is the lowest certainty level \( \lambda \) in
\([0,1]\) for which the classical theory \( \Theta_\lambda \) is consistent. An inconsistent-tolerant inference relation \( \vdash_{\text{poss}} \) for possibilis-
tic logic can then be defined as follows:

\[
\Theta \vdash_{\text{poss}} \alpha \iff \Theta_{inc(\alpha)} \vdash \alpha
\]

We will write \( (\Theta, \alpha) \vdash_{\text{poss}} \beta \) as an abbreviation for \( \Theta \cup \{(\alpha, 1)\} \vdash_{\text{poss}} \beta \). It can be shown that \( \Theta \vdash_{\text{poss}} (\alpha, \lambda) \) can be
decided by making \( O(\log_2 k) \) calls to a SAT solver, with \( k \) the number of certainty levels in \( \Theta \) [Lang, 2001].

There is a close relationship between possibilistic logic and the rational closure of a set of defaults. Recall that \( \alpha \vdash \beta \) is
tolerated by a set of defaults \( \{\alpha_1 \vdash \beta_1, ..., \alpha_n \vdash \beta_n\} \) if the classical formula \( \alpha \land \beta \land \bigwedge \neg (\alpha_i \lor \beta_i) \) is consistent [Pearl, 1990]. Let \( \Delta \) be a set of defaults. The rational closure of \( \Delta \) is based on a stratification \( \Delta_1, ..., \Delta_k \), known as the Z-ordering, where each \( \Delta_j \) contains all defaults from \( \Delta \setminus (\Delta_1 \cup ... \Delta_{j-1}) \) which are tolerated by \( \Delta \setminus (\Delta_1 \cup ... \Delta_{j-1}) \). Intuitively, \( \Delta_1 \) contains the most general default rules, \( \Delta_2 \) contains exceptions to these rules, \( \Delta_3 \) contains exceptions to these exceptions, etc. Given the stratification \( \Delta_1, ..., \Delta_k \) we define the
possibilistic logic theory \( \Theta = \{(\neg \alpha \lor \beta, \lambda_i) | (\alpha \vdash \beta) \in \Delta_i\} \), where we assume \( 0 < \lambda_1 < ... < \lambda_k < 1 \). It then holds that \( \alpha \vdash \beta \) is in the rational closure of \( \Delta \) iff \( (\Theta, \alpha) \vdash_{\text{poss}} \beta \) [Benferhat et al., 1998].

3.2 Learning Theory

We now cover some basic notions from statistical learn-
ing theory [Vapnik, 1995]. We restrict ourselves to bi-
ary classification problems, where the two labels are
1 and -1. Let \( X \) be a set of examples. A hypothesis is a
function \( h : X \to \{-1,1\} \). A hypothesis \( h \) is said to
cover an example \( e \in X \) if \( h(e) = 1 \). Consider a set \( S \subseteq X \times \{-1,1\} \) of \( n \) labeled examples that have been
squared from a distribution \( p \). A hypothesis \( h \)'s sample error rate is \( err_S(h, S) = \frac{1}{n} \sum_{(x,e) \in S} 1(h(x) \neq e) \) where
\( 1(\alpha) = 1 \) if \( \alpha \equiv true \) and \( 1(\alpha) = 0 \) otherwise. A hypo-
thesis \( h \)'s expected error w.r.t. the probability distribution
\( p \) is given by \( err_p(h) = E_{(x,e) \sim p} [1(h(x) \neq e)] \). Statis-
tical learning theory provides tools for bounding the prob-
ability \( P(\sup_{h \in H} |err_p(h) - err_S(h, S)| \geq \epsilon) \), where \( S \) is
known to be sampled iid from \( p \) but \( p \) itself is unknown. These
bounds link \( h \)'s training set error to its (probable) per-
formance on other examples drawn from the same distribu-
tion, and therefore permits theoretically controlling overfit-
ting. The most important bounds of this type depend on the
Vapnik-Chervonenkis (VC) dimension [Vapnik, 1995].

**Definition 1** (Vapnik-Chervonenkis (VC) dimension). A hy-
pothesis set \( H \) is said to shatter a set of examples \( Y \) if for
every subset \( Z \subseteq Y \) there is a hypothesis \( h \in H \) such that
\( h(e) = 1 \) for every \( e \in Z \) and \( h(e) = -1 \) for every \( e \in Y \setminus Z \).

The VC dimension of \( H \) is the cardinality of the largest set
that is shattered by \( H \).

Upper bounds based on the VC dimension are increasing functions of the VC dimension and decreasing functions of the number of examples in the training sample \( S \). Ideally, the goal is to minimize expected error, but this cannot be evaluated since \( p \) is unknown. Structural risk minimization [Vapnik, 1995] helps with this if the hypothesis set can be organized into a hierarchy of nested hypothesis classes of in-
creasing VC dimension. It suggests selecting hypotheses that
minimize a risk composed of the training set error and a com-
plexity term, e.g. if two hypotheses have the same training set
error, the one originating from the class with lower VC
dimension should be preferred.

4 Learning from Default Rules

In this section, we formally describe a new learning setting for possibilistic logic called learning from default rules. We
assume a finite alphabet \( \Sigma \) is given. An example is a default rule over \( \Sigma \) and a hypothesis is a possibilistic logic theory over \( \Sigma \). A hypothesis \( h \) predicts the class of an example \( e = \alpha \vdash \beta \) by checking if \( h \) covers \( e \), in the following sense.

**Definition 2** (Covering). A hypothesis \( h \in H \) covers an ex-
ample \( e = \alpha \vdash \beta \) if \( (h, \alpha) \vdash_{\text{poss}} \beta \).

The hypothesis \( h \) predicts positive, i.e. \( h(\alpha \vdash \beta) = 1 \), iff \( h \)
covers \( e \), and else predicts negative, i.e. \( h(\alpha \vdash \beta) = -1 \).
Example 1. Let us consider the following set of examples
\[ S = \{ (\text{bird} \land \text{antarctic} \not\rightarrow \text{flies}, 1), (\text{bird} \not\rightarrow \text{flies}, -1) \} \]
The following hypotheses over the alphabet \{bird, flies, antarctic\} cover all positive and no negative examples:
\[ h_1 = \{ (\text{bird}, 1), (\text{antarctic} \rightarrow \text{flies}, 1) \} \]
\[ h_2 = \{ (\text{flies}, 0.5), (\text{antarctic} \rightarrow \text{flies}, 1) \} \]
\[ h_3 = \{ (\text{antarctic} \not\rightarrow \text{flies}, 1) \} \]
The learning task can be formally described as follows.

Given: A multi-set \( S \) which is an i.i.d. sample from a set of default rules over a given finite alphabet \( \Sigma \).

Do: Learn a possibilistic logic theory that covers all positive examples and none of the negative examples in \( S \).

The above definition assumes that \( S \) is perfectly separable, i.e. it is possible to perfectly distinguish positive examples from negative examples. In practice, we often relax this requirement, and instead aim to find a theory that minimizes the training set error. Similar to learning in graphical models, this learning task can be decomposed into parameter learning and structure learning. In our context, the goal of parameter learning is to convert a set of propositional formulas into a possibilistic logic theory, while the goal of structure learning is to decide what that set of propositional formulas should be.

4.1 Parameter Learning

Parameter learning assumes that the formulas of the possibilistic logic theory are fixed, and only the certainty weights need to be assigned. As the exact numerical values of the certainty weights are irrelevant, we will treat parameter learning as the process of finding the most suitable stratification of a given set of formulas, e.g. the one which minimizes training error or structural risk (cf. Section 4.2).

Example 2. Let \( S = \{ (\text{penguin} \not\rightarrow \text{bird}, 1), (\text{bird} \not\rightarrow \text{flies}, 1), (\text{penguin} \not\rightarrow \text{flies}, 1), (\text{not} \text{bird}, -1), (\text{not} \text{penguin}, -1) \} \) and \( T = \{ \text{bird}, \text{flies}, \text{penguin}, \text{not} \text{penguin} \lor \text{not} \text{flies} \} \). A stratification of \( T \) which minimizes the training error on the examples from \( S \) is \( T^+ = \{ \text{bird}, 0.25 \}, (\text{penguin}, 0.25), (\text{flies}, 0.5), (\text{penguin} \lor \neg \text{flies}) \} \) which is equivalent to \( T^{**} = \{ \text{flies}, 0.5 \}, (\neg \text{penguin} \lor \neg \text{flies}, 1) \} \) because \( inc(T^*) = 0.25 \). Note that \( T^{**} \) correctly classifies all examples except \( (\text{penguin} \not\rightarrow \text{bird}) \).

Given a set of examples \( S \), we write \( S^+ = \{ \alpha | (\alpha, 1) \in S \} \) and \( S^- = \{ \alpha | (\alpha, -1) \in S \} \). A stratification \( T^* \) of a theory \( T \) is a separating stratification of \( S^+ \) and \( S^- \) if it covers all examples from \( S^+ \) and no examples from \( S^- \).

Example 3. Let us consider the following set of examples \( S = \{ (\neg x, 1), (\neg y, 1), (x \not\rightarrow a, 1), (y \not\rightarrow b, 1), (x \land y \not\rightarrow a, -1) \} \). Let \( T = \{ \neg x, \neg y, \neg x \lor a, \neg y \lor b \} \). The following stratification is a separating stratification of \( S^+ \) and \( S^- : h = \{ (\neg x, 0.25), (\neg y, 0.85), (\neg y, 0.75), (\neg y \lor b, 1) \} \)\). Note that the Z-ranking of \( S^+ \) also corresponds to a stratification of \( T \), as \( T \) contains exactly the clause representations of the positive examples. However, using the Z-ranking leads to a different stratification, which is: \( h_z = \{ (\neg x, 0.5), (\neg y, 0.5), (\neg x \lor a, 1), (\neg y \lor b, 1) \} \). Note that \( h_z(x \land y \not\rightarrow a) = 1 \) whereas \( h(x \land y \not\rightarrow a) = -1 \).

Because arbitrary stratifications can be chosen, there is substantial freedom to ensure that negative examples are not covered. This is true even when the set of considered formulas is restricted to the clause representations of the positive examples, as seen in Example 3. Unfortunately, the problem of finding an optimal stratification is computationally hard.

Theorem 1. Deciding whether a separating stratification exists for given \( T, S^+ \) and \( S^- \) is a \( \Sigma_2^1 \)-complete problem.

Proof. The proof of the membership result is trivial. We show the hardness result by reduction from the \( \Sigma_2^1 \)-complete problem of deciding the satisfiability of quantified Boolean formulas of the form \( \exists X \forall Y: \Phi(X,Y) \) where \( X \) and \( Y \) are vectors of propositional variables and \( \Phi(X,Y) \) is a propositional formula. Let \( T = X \cup \{ \neg x: x \in X \} \cup \{ \Phi(X,Y) \rightarrow aux \} \) be a propositional theory, let \( S^+ = \{ \neg aux \} \) and \( S^- = \emptyset \). We need to show that \( \exists X \forall Y: \Phi(X,Y) \) is satisfiable if and only if there exists a separating stratification for \( T \), \( S^+ \) and \( S^- \). Let \( \theta \) be an assignment of variables in \( X \) such that \( \forall Y: \Phi(X,Y, \theta) \) is true. Then we can construct the separating stratification as
\[
(\{ \Phi(X,Y) \rightarrow aux \} \cup \{ x \in X : x \theta = 1 \} \cup \{ \neg x: x \in X \land x \theta = 0 \}) \cup \{ \neg x: x \in X \land x \theta = 0 \} \cup \{ x \in X : x \theta = 0 \}.
\]

Since \( \Phi(X,Y) \) will always be true in any model consistent with the highest level of the stratification, because of the way we chose \( x \) and \( \neg x \) for this level, so will \( aux \). \( \Rightarrow \) Let \( T^* \) be a stratification of \( T \) which entails the default rule \( \neg aux \). We can assume w.l.o.g. that \( T^* \) has only two levels. Since \( T^* \) is a separating stratification, we must have \( T^* \vdash \text{aux} \). Therefore the highest level \( L^* \) of \( T^* \) must be a consistent theory and \( \Phi(X,Y) \) must be true in all of its models. Let \( X' = \{ x \in X : x \in L^* \lor \neg x \in L^* \} \) and \( X'' = X \setminus X' \). We can construct an assignment \( \theta \) to variables in \( X' \) by setting \( x \theta = 1 \) for \( x \in L^* \) and \( x \theta = 0 \) for \( \neg x \in L^* \). It follows from the construction that \( \forall X' \forall Y : \Phi(X,Y, \theta) \) must be true.

As this result reveals, in practice we will need to rely on heuristic methods for parameter learning. In Section 4.3 we will propose such a heuristic method, which will moreover also include structure learning.

4.2 VC Dimension of Possibilistic Logic Theories

We explore the VC dimension of the set of possible stratifications of a propositional theory, as this will allow us to provide probabilistic bounds on the generalization ability of a learned possibilistic logic theory. Let us write Strat(\( T \)) for the set of all stratifications of a propositional theory \( T \), and let Strat(\( k \)) \( (T) \) be the set of all stratifications with at most \( k \) levels. The following proposition provides an upper bound for the VC dimension and can be proved by bounding the cardinality of Strat(\( k \))(\( T \)).

Proposition 1. Let \( T \) be a set of \( n \) propositional formulas. Then \( VC(\text{Strat}(k)(T)) \leq n \log_2 k \).

In the next theorem, we establish a lower bound on the VC dimension of stratifications with at most \( k \) levels which shows that the above upper bound is asymptotically tight.
Theorem 2. For every $k, n, k \leq n$, there is a propositional theory $T$ consisting of $n$ formulas such that

$$VC(\text{Strat}^{(k)}(T)) \geq \frac{1}{4} n (\log_2 k - 1).$$

To prove Theorem 2, we need the following lemmas; some straightforward proofs are omitted due to space constraints.

Lemma 1. If $S$ is a totally ordered set, let $kth(S, i)$ denote the $i$-th highest element of $S$. Let $X = \{x_1, \ldots, x_n\}$ be a set of cardinality $n = 2^m$ where $m \in \mathbb{N} \setminus \{0\}$. Let

$$C = \{x_1 < x_2, x_3 < x_4, \ldots, x_{n-1} < x_n, \ kth(x_1, x_2), 1 < kth(x_3, x_4), 1, \ kth(x_1, x_2), 2 < kth(x_3, x_4), 2, \ kth(x_5, x_6), 1 < kth(x_7, x_8), 1, \ kth(x_5, x_6), 2 < kth(x_7, x_8), 2, \ldots\}$$

be a set of $\frac{1}{2} n \log_2 n$ inequalities. Then for any $C' \subseteq C$ there is a permutation of $X$ satisfying all constraints from $C'$ and no constraints from $C \setminus C'$. Let $\mathcal{X}$ be a propositional logic theory. Let $\mathcal{Y} = \{y_1, \ldots, y_m\}$ and $\mathcal{Z} = \{z_1, \ldots, z_m\}$ be disjoint subsets of $X$. Then

$$(\mathcal{T}^*, \text{at-least}_{m_y-k+1} (\neg y_1, \neg y_2, \ldots, \neg y_{m_y})) \vdash_{\text{poss}} \text{at-least}_{m_z} (z_1, z_2, \ldots, z_{m_z})$$

iff $kth(y_1, \ldots, y_{m_y}, k) < kth(z_1, \ldots, z_{m_z}, k)$ w.r.t. the ordering given by the permutation $\pi$.

Lemma 3. For every $n = 2^m$ there is a propositional theory $\mathcal{T}$ consisting of $n$ formulas such that

$$VC(\text{Strat}(\mathcal{T})) \geq \frac{1}{2} n \log_2 n.$$

Proof. Let $\mathcal{T} = \{x_1, \ldots, x_n\}$ be a set of propositional variables where $n = 2^m$, $m \in \mathbb{N} \setminus \{0\}$ and let $C$ be defined as in Lemma 1. Let $D = \{\text{at-least}_{m_y-k+1} (\neg x_i, \ldots, \neg x_i) \mid \text{at-least}_{m_z} (x_j, \ldots, x_j)\}$

$$(kth(x_i, \ldots, x_i), k) < kth(x_j, \ldots, x_j), k) \in C\}$$

i.e. $D$ contains one default rule for every inequality from $C$. It follows from Lemma 1 and Lemma 2 that the set $D$ can be shattered by stratifications of the propositional theory $\mathcal{T}$. The cardinality of $D$ is $\frac{1}{2} n \log_2 n$. Therefore the VC dimension of stratifications of $\mathcal{T}$ is at least $\frac{1}{2} n \log_2 n$. \qed

Proof of Theorem 2. We show that if $k$ and $n$ are powers of two then $\frac{1}{2} n \log_2 k$ is a lower bound of the VC dimension. The general case of the theorem then follows straightforwardly. Let $T = \bigcup_{i=1}^{\log_2 k} \{x_{(i-1)k+1}, \ldots, x_{ik}\}$ and let $D_i$ be a set of default rules of cardinality $\frac{1}{2} n \log_2 k$ shattered by $\text{Strat}(\{x_{(i-1)k+1}, \ldots, x_{ik}\})$. It follows from Lemma 3 that such a set $D_i$ always exists. Let $D = \bigcup_{i=1}^{\log_2 k} D_i$. Then $D$ has cardinality $\frac{1}{2} n \log_2 k$ and is shattered by $\text{Strat}^{(k)}(T)$. To see that the latter holds, note that the sets of formulas $\text{Strat}(\{x_{(i-1)k+1}, \ldots, x_{ik}\})$ are disjoint. Therefore, if we want to find a stratification from $\text{Strat}^{(k)}(T)$ which covers only examples from an arbitrary set $D' \subseteq D$ and no other examples from $D$ then we can merge stratifications of $\{x_{(i-1)k+1}, \ldots, x_{ik}\}$ which cover exactly the examples from $D_i \cap D'$, where merging stratifications is done by levelwise unions. \qed

Combining the derived lower bounds and upper bounds on the VC dimension together with the structural risk minimization principle, we find that given two stratifications with the same training set error rate, we should prefer the one with the fewest levels. Furthermore, when structure learning is used, it is desirable for learned theories to be compact. A natural learning problem then consists of selecting a small subset of $\mathcal{T}$, where $\mathcal{T}$ corresponds to the set of formulas considered by the structure learner, and identifying a stratification only for that subset. The results in this section can readily be extended to provide bounds on the VC dimension of this problem. Let $\mathcal{T}$ be a propositional theory of cardinality $n$ and let $m < n$ be a positive integer. The VC dimension of the set of hypotheses involving at most $m$ formulas from $\mathcal{T}$ and having at most $k$ levels is bounded by $m (\log_2 n + \log_2 k)$. This can simply be obtained by upper-bounding the number of the different stratifications with at most $k$ levels and $m$ formulas selected from a set of cardinality $n$, by $n^m \cdot k^m$.

4.3 Heuristic Learning Algorithm

In this section, we propose a practical heuristic algorithm for learning a propositional logic theory from a set $S$ of positive and negative examples of default rules. Our method combines greedy structure learning with greedy weight learning. We assume that every default or non-default $\alpha \vdash \beta$ in $S$ is such that $\neg \alpha$ and $\beta$ correspond to clauses.

The algorithm starts by initializing the “working” stratification $\mathcal{T}^*$ to be an empty list. Then it repeats the following revision procedure for a user-defined number of iterations $n$, or until a timeout is reached. First, it generates a set of candidate propositional clauses $C$ as follows:

- It samples a set of defaults $\alpha \vdash \beta$ from the examples that are misclassified by $\mathcal{T}^*$.
- For each default $\alpha \vdash \beta$ which has been sampled, it samples a subclause $\neg \alpha'$ of $\neg \alpha$ and a subclause $\beta'$ of $\beta$. If $\alpha \vdash \beta$ is a positive example then $\neg \alpha' \lor \beta'$ is added to $C$; if it is a negative example, then $\neg \alpha' \lor \beta''$ is added instead, where $\beta''$ is obtained from $\beta'$ by negating each of the literals.


The algorithm then tries to add each formula in C to an existing level of $T^*$ or to a newly inserted level. It picks the clause $c$ whose addition leads to the highest accuracy and adds it to $T^*$. The other clauses from C are discarded. In case of ties, the clause which leads to the stratification with the fewest levels is selected, in accordance with the structural risk minimization principle and our derived VC dimension. If there are multiple such clauses, then it selects the shortest among them. Subsequently, the algorithm tries to greedily minimize the newly added clause $c$, by repeatedly removing literals as long as this does not lead to an increase in the training set error. Next, the algorithm tries to revise $T^*$ by greedily removing clauses whose deletion does not increase the training set error. Finally, as the last step of each iteration, the weights of all clauses are optimized by greedily reinserting each clause in the theory.

5 Experiments

We evaluate our heuristic learning algorithm in two different applications: learning domain theories from crowdsourced default rules and approximating MAP inference in propositional Markov logic networks. As we are not aware of any existing methods that can learn a consistent logical theory from a set of noisy defaults, there are no baseline methods to which our method can directly be compared. However, if we fix a target literal $l$, we can train standard classifiers to predict for each propositional context $\alpha$ whether the default $\alpha \vdash l$ holds. This can only be done consistently with “parallel” rules, where the literals in the consequent do not appear in antecedents. We will thus compare our method to three traditional classifiers on two crowdsourced datasets of parallel rules: random forests [Breiman, 2001], C4.5 decision trees [Quinlan, 1993], and the rule learner RIPPER [Cohen, 1995]. Random forests achieve state-of-the-art accuracy but its models are difficult to interpret. Decision trees are often less accurate but more interpretable than random forests. Finally, rule learners have the most interpretable models, but often at the expense of lower accuracy. In the second experiment, approximating MAP inference, we do not restrict ourselves to parallel rules. In this case, only our method can guarantee that the predicted defaults will be consistent.

5.1 Methodology

Our learning algorithm is implemented in Java and uses the SAT4J library [Berre and Parrain, 2010]. The implementation contains a number of optimizations which make it possible to handle datasets of thousands of default rules, including caching, parallelization, detection of relevant possibilistic subtheories for deciding entailment queries and unit propagation in the possibilistic logic theories.

We use the Weka [Hall et al., 2009] implementations for the three baselines. When using our heuristic learning algorithm, we run it for a maximum time of 10 hours for the crowdsourcing experiments reported in Section 5.2 and for one hour for the experiments reported in Section 5.3. For C4.5 and RIPPER, we use the default settings. For random forests, we used the default settings and set the number of trees to 100.

5.2 Learning from Crowdsourced Examples

We used CrowdFlower, an online crowdsourcing platform, to collect expert rules about two domains. In the first experiment, we created 3706 scenarios for a team on offense in American football by varying the field position, down and distance, time left, and score difference. Then we presented six choices for a play call (punt, field goal, run, pass, kneel down, do not know/it depends) and asked the user to select the most appropriate one. All scenarios were presented to 5 annotators. A manual inspection of a subset of the rules revealed that they are of reasonably high quality. In a second experiment, users were presented with 2388 scenarios based on Texas hold’em poker situations, where users were asked whether in a given situation they would typically fold, call or raise, with a fourth option again being “do not know/it depends”. Each scenario was again presented to 5 annotators. Given the highly subjective nature of poker strategy, it was not possible to enforce the usual quality control mechanism on CrowdFlower in this case, and the quality of the collected rules was accordingly found to be more variable.

In both cases, the positive examples are the rules obtained via crowdsourcing, while negative examples are created by taking positive examples and randomly selecting a different consequent. To create training and testing sets, we divided the data based on annotator ID so that all rules labeled by a given annotated appear only in the training set or only in the testing set, to prevent leakage of information. We added a set of hard rules to the possibilistic logic theories to enforce that only one choice should be selected for a game situation. The baseline methods were presented with the same information, in the sense that the problem was presented as a multi-class classification problem, i.e. given a game situation, the different algorithms were used to predict the most typical action (with one additional option being that none of the actions is typical). The results are summarized in Table 1.

In the poker experiment, our approach obtained slightly higher accuracy than random forest and RIPPER but performed slightly worse than C4.5. However, a manual inspection showed that a meaningful theory about poker strategy was learned. For example, at the lowest level, the possibilistic logic theory contains the rule “call”, which makes sense given the nature of the presented scenarios. At a higher level, it contains more specific rules such as “if you have three of a kind then raise”. At the level above, it contains exceptions to these more specific rules such as “If you have three of a kind, there are three hearts on the board and your opponent raised on the river then call”. In the American football experiment, our approach obtained lower accuracy than the competing algorithms. The best accuracy was achieved by C4.5. Again, we also manually inspected the learned possibilistic logic theory and found that it captures some general intuitions and known strategy about the game. For example, the most general rule is “pass”

2The data, code, and learned models are available from [https://github.com/supertweety/](https://github.com/supertweety/).

3A recent large-scale empirical evaluation has shown that variants of the random forest algorithm tend to perform best on real-life datasets [Fernández-Delgado et al., 2014].
5.3 Approximating MAP Inference

Markov logic networks can be seen as weighted logical theories. The weights assigned to formulas are intuitively seen as penalties; they are used to induce a probability distribution over the set of possible worlds. Here we are interested in maximum a posteriori (MAP) inference. Specifically, we consider entailment relations from \( \Theta \) over the set of possible worlds. Here we are interested in maximizing a posteriori (MAP) inference, which could e.g. easily be combined with expert knowledge.

Table 1: Test set accuracies.

<table>
<thead>
<tr>
<th></th>
<th>Poss.</th>
<th>Rand. F.</th>
<th>C4.5</th>
<th>RIPPER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Poker</td>
<td>40.5</td>
<td>38.6</td>
<td>41.1</td>
<td>39.9</td>
</tr>
<tr>
<td>Football</td>
<td>68.3</td>
<td>72.4</td>
<td>74.6</td>
<td>73.1</td>
</tr>
<tr>
<td>NLTCS</td>
<td>78.1</td>
<td>69.6</td>
<td>70.2</td>
<td>67.7</td>
</tr>
<tr>
<td>MSNBC</td>
<td>62.0</td>
<td>61.9</td>
<td>62.0</td>
<td>48.8</td>
</tr>
<tr>
<td>Plants</td>
<td>73.1</td>
<td>77.8</td>
<td>71.4</td>
<td>53.8</td>
</tr>
<tr>
<td>DNA</td>
<td>52.8</td>
<td>56.6</td>
<td>54.9</td>
<td>51.1</td>
</tr>
</tbody>
</table>

which is the most common play type. Another example is that second most general level has several rules that say on fourth down and long you should punt. More specific levels allow for cases when you should not punt on fourth down, such as when you are in field goal range.

Despite not achieving the same accuracy as C4.5 in this experiment, it nonetheless seems that our method is useful for building up domain theories by crowdsourcing opinions. The learned domain theories are easy to interpret (e.g., the size of the poker theory, as a sum of rule lengths, is more than 10 times smaller than the number of nodes in the learned tree) and capture relevant strategies for both games. The models obtained by classifiers such as C4.5, on the other hand, are often difficult to interpret. Moreover, traditional classifiers such as C4.5 can only be applied to parallel rules, and will typically lead to inconsistent logical theories in more complex domains. In contrast, our method can cope with arbitrary default rules as input, making it much more broadly applicable for learning domain theories.

6 Conclusions

The aim of this paper was to study the problem of reasoning with default rules from a machine learning perspective. We have formally introduced the problem of learning from defaults and have analyzed its theoretical properties. Among others, we have shown that the complexity of the main decision problem is \( \Sigma^P_2 \) complete, and we have established asymptotically tight bounds on the VC-dimension. At the practical level, we have proposed practical heuristic learning algorithms, which can scale to datasets with thousands of rules. We have presented experimental results that show the application potential of the proposed learning algorithm, considering two different application settings: learning domain theories by crowdsourcing expert opinions and approximating propositional MLNs. We believe that the methods proposed in this paper will open the door to a wider range of applications of default reasoning, where we see defaults as a convenient interface between experts and learned domain models.
Acknowledgments

This work has been supported by a grant from the Leverhulme Trust (RPG-2014-164). Jesse Davis is partially supported by the KU Leuven Research Fund (C22/15/015), and FWO-Vlaanderen (G.0356.12, SBO-150033).

References


