Satisfiability Machines

Filip Železný

Czech Technical University in Prague, Faculty of Electrical Engineering,
Department of Cybernetics, Intelligent Data Analysis Research laboratory

Abstract. We propose a probabilistic model for formulas, in which a formula's probability decreases with the number of pre-defined constraints it contradicts. Probability of ground conjunctions can be computed by a series of subsumption checks. The probability is equivalent (up to a multiplicative constant) to that computed by a Markov Logic Network for conjunctions that fully describe a possible world. An experiment indicates that the two quantities correlate also for other conjunctions, with less variance for larger conjunctions. The proposed framework suggests a simple classification principle.

1 Introduction

In Markov Logic Networks (MLN) [6], *the probability of an interpretation (possible world) decreases with the number of pre-defined constraints it violates*. The probability of a formula is then the sum of probabilities of the worlds in which it holds. This quantity is intractable to calculate and sampling is usually adopted to approximate it. Here we explore a simplified probabilistic model enabling fast computation of a formula's probability. We embrace the key concept of MLN's but formalize it without regards to possible worlds: *the probability of a formula decreases with the number of pre-defined constraints it contradicts*. Instead of general formulas, we focus here specifically on ground conjunctions. Having a parametric probability model for ground conjunctions can be useful since they often represent learning examples (they correspond to partial interpretations [1]). In the sequel we explore how such probabilities can be computed, how they relate to probabilities in the MLN framework, and how our framework can be used for classification.

2 Probabilistic Model

We consider a first-order logic language with Herbrand base (set of all ground atoms in the language) $\mathcal{H}$. As in most treatments of MLN's, we do not allow functions in the language so that $\mathcal{H}$ is finite. Extensions to the infinite case (e.g. along the lines of [8]) are possible but not dealt with here. Given a finite set of formulas $F_1, F_2, \dots F_n$ with real weights $w_1, w_2, \dots w_n$, the probability of an interpretation $x \subseteq \mathcal{H}$ in the MLN framework is

$$P_M(x) = \frac{1}{Z_M} \exp \sum_{i=1}^{n} w_i n_i(x) \tag{1}$$

where n_i is the number of groundings of F_i that are true in x, i.e., denoting the set of all its grounding substitutions possible in the language as $\mathsf{GS}(F_i)$,

$$n_i(x) = |\{\theta \in \mathsf{GS}(F_i) \mid x \models F_i\theta\}| \tag{2}$$

Z_M is a normalizer ensuring that the above probability sums to 1 over all $2^{|\mathcal{H}|}$ interpretations x. As in most MLN studies, we shall assume that the F_i's are clauses.

The probability of a formula φ in the MLN framework is the sum of probabilities of φ's models

$$P_M(\varphi) = \sum_{x \models \varphi} P_M(x) = \frac{1}{Z_M} \sum_{x \models \varphi} \exp \sum_{i=1}^{n} w_i n_i(x) \tag{3}$$

Computing this probability is not tractable except for minimalistic languages, and in practice it is estimated through sampling. Particular attention has been devoted to approximative methods for the case when $\varphi = c$ is a ground conjunction [6]. Here we instead aim at a conceptual simplification. Specifically, we maintain the F_i's and w_i's but we propose that

$$P_S(\varphi) = \frac{1}{Z_S} \exp \sum_{i=1}^{n} w_i m_i(\varphi) \tag{4}$$

where

$$m_i(\varphi) = |\{\theta \in \mathsf{GS}(F_i) \mid \varphi \wedge F_i\theta \nvdash \bot\}| \tag{5}$$

So the probability of a formula decreases with the number of constraints (groundings of the F_i's) it contradicts. Calculating the normalizing constant Z_S is less obvious than calculating Z_M in the MLN case, however, it becomes straightforward if we again constrain ourselves to $\varphi = c$ being ground conjunctions (we will maintain the assumption throughout the paper). Then Z_S is such that $P_S(c)$ sums to 1 over all $3^{|\mathcal{H}|}$ possible ground conjunctions c (each atom from $\mathcal{H}$ can either be positive, negative, or absent in c).

Note that $c \wedge F_i\theta \vdash \bot$ is equivalent to $F_i\theta \vdash \neg c$. Since c is a ground conjunction, $\neg c$ is a ground clause and thus, unless F_i is self-resolving, $F_i\theta \vdash \neg c$ can be reduced to the subsumption check $\mathsf{lits}(F_i\theta) \subseteq \mathsf{lits}(\neg c)$. Eq. 5 can now be expressed as

$$m_i(c) = |\{\theta \in \mathsf{GS}(F_i) \mid \mathsf{lits}(F_i\theta) \nsubseteq \mathsf{lits}(\neg c)\}| \tag{6}$$

Equivalently, $m_i(c) = |\mathsf{GS}(F_i)| - |\{\theta \in \mathsf{GS}(F_i) \mid \mathsf{lits}(F_i\theta) \subseteq \mathsf{lits}(\neg c)\}|$. The first summand is not significant since it does not depend on c and can simply be removed and reflected rather in the weight w_i. Thus, what remains to compute towards obtaining $m_i(c)$ is just the set of solutions to a subsumption check. This is particularly appealing due to extremly fast subsumption testers under energetic research now that achieve order-of-magnitude speed-ups by employing

CSP algorithms [5] or other heuristic approaches [7], language-biases [4] or randomized search [3,2]. The approach [2] seems particularly relevant in the present context since it uses randomization to estimate the *number* of subsumption solutions, i.e, the $m_i(c)$ number directly.

3 Comparing $P_M(c)$ and $P_S(c)$

The relationship between $P_M(c)$ and $P_S(c)$ is clear for *model conjunctions* (conjunctions that have a single model). The model conjunction for an interpretation x is $c_x = \bigwedge_{a \in x} a \bigwedge_{b \in \mathcal{H} \setminus x} \neg b$. We have that

$$Z_S P_S(c_x) = Z_M P_M(c_x) \tag{7}$$

for any model conjunction c_x. Comparing Eqs. 4 and Eq. 3, we see that the above is true if

$$\exp \sum_{i=1}^{n} w_i m_i(c_x) = \sum_{y \models c_x} \exp \sum_{i=1}^{n} w_i n_i(y)$$

However, since the only model of c_x is x, the first summation on the right-hand side vanishes and we only need to check that for each i, $m_i(c_x) = n_i(x)$. As follows from Eqs. 5 and 2, this is the case if the relation $c_x \wedge F_i\theta \nvdash \bot$, i.e., $\mathsf{lits}(F_i\theta) \nsubseteq \mathsf{lits}(\neg c_x)$ is equivalent to the relation $x \models F_i\theta$. Since $\neg c_x = \bigvee_{a \in x} \neg a \bigvee_{b \in \mathcal{H} \setminus x} b$, the former relation means that either some negative literal of $F_i\theta$ is not in x or some positive literal of $F_i\theta$ is not in $\mathcal{H} \setminus x$, i.e. (since $x \subseteq \mathcal{H}$), is in x. That, however, is precisely the definition of the latter relation and we have thus proven the equivalence of both relations. Thus Eq. 7 indeed holds.

For ground conjunctions c other than model conjunctions, the relationship between $P_S(c)$ and $P_M(c)$ is less clear. For this general case, we have not yet been able to relate the quantities analytically as in Eq. 7 and so we approached the question empirically. To this end, we implemented in YAP Prolog both Markov Logic Networks inference, and the herewith proposed subsumption-based inference, both exact and sampling-based.[1]

For the particular experiment, we considered the following four clauses from a toy university domain, each with weight $w_i = 1$

$$F_1 = \text{false} \leftarrow \text{studies}(P, C) \wedge \text{teaches}(P, C) \tag{8}$$

$$F_2 = \text{false} \leftarrow \text{teaches}(\mathsf{john}, C) \wedge \text{teaches}(\mathsf{jack}, C) \tag{9}$$

$$F_3 = \text{teaches}(\mathsf{john}, C) \vee \text{teaches}(\mathsf{jack}, C) \leftarrow \text{studies}(P, C) \tag{10}$$

$$F_4 = \text{studies}(\mathsf{john}, C) \vee \text{studies}(\mathsf{jack}, C) \leftarrow \text{teaches}(P, C) \tag{11}$$

We used a *typed* Herbrand base $\mathcal{H}$, which consists of atoms made of one of teaches/2 and studies/2 predicate symbols with the first argument being either jack or john and the second argument being one of $\{\mathsf{math}, \mathsf{cs}, \mathsf{ai}\}$. $\mathcal{H}$ contained

[1] The implementation (sporadically commented) is available at `http://labe.felk.cvut.cz/~zelezny/sm.yap`.

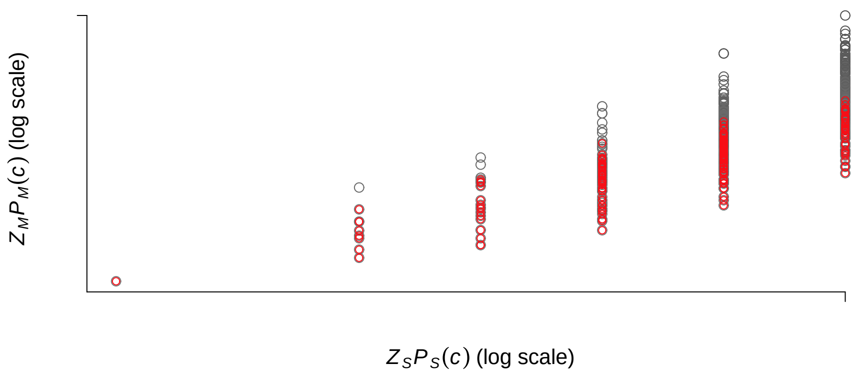

Fig. 1. MLN-probabilities vs. SM-probabilities of random ground conjunctions. The mean length of a conjunction is 6 literals, conjunctions of this or greater length are shown in red.

exactly 12 atoms. We randomly generated 500 ground conjunctions. Each one was sampled independently from the others, by going over all atoms in $\mathcal{H}$. Each atom in $\mathcal{H}$ had 1/2 probability of being included in the conjunction; if it was included, it produced a positive (negative) literal with 1/2 probability. For each such conjunction c we computed both $Z_M P_M(c)$ and $Z_S P_S(c)$. The 500 value pairs are shown diagrammatically in Fig. 1. The main insight provided by the experiment is that the two quantities are clearly correlated and the satisfiability framework is characterized by a rougher probability scale. As expected, the conditional variance $var(P_M(c)|P_S(c))$ is smaller for large conjunctions c (red marks in the figure) than for arbitrary conjunctions; this is because large conjunctions are closer to model conjunctions for which we have already established the deterministic relationship (7).

While $P_S(c)$ has its own intuitive meaning (as worded in introducing the quantity), the experiment shows that it can be also viewed as an estimator of $P_M(c)$. Note that unlike $P_M(c)$, $P_S(c)$ is reasonable for modeling only those domains where the constraints F_i are independent of each other; otherwise it may fail intuition. For example, given $F_1 = p \leftarrow q$, $F_2 = q \leftarrow r$, and $c = r \wedge \neg p$, then $P_S(c)$ is a mode of the P_S distribution since c does not violate any of the two constraints. The fact that it contradicts their conjunction is not reflected by $P_S(c)$. We think, however, that the said independence assumption is not problematic for modeling real-world domains, as essentially the same assumption is made by the generally successful propositionalization systems.

The natural question is what benefits $P_S(c)$ brings us in comparison to $P_M(c)$. The key advantage lies in the fact that the exact computation of $Z_M P_M(c)$ takes time exponential in the size of $\mathcal{H}$ and this size in turn grows combinatorially with the number of constants in the language. To appreciate this complexity, we calculated $Z_M P_M(c)$ for the (randomly drawn) conjunction $c = \neg\text{teaches}(\mathsf{john}, \mathsf{math}) \wedge \text{teaches}(\mathsf{john}, \mathsf{ai}) \wedge \neg\text{teaches}(\mathsf{jack}, \mathsf{ai}) \wedge \neg\text{studies}(\mathsf{john}, \mathsf{math}) \wedge \neg\text{studies}(\mathsf{jack}, \mathsf{math}) \wedge \neg\text{studies}(\mathsf{jack}, \mathsf{ai})$ for the previously shown F_i's and for different numbers of constants in $\mathcal{H}$, obtaining the following runtimes

number of course-constants in $\mathcal{H}$	2	3	4	5
time to compute $Z_M P_M(c)$ [s]	0.02	0.47	10.25	242.67

In contrast, the time to exactly calculate $Z_S P_S(c)$ is zero (below measurability) in all these cases. This is understandable since the runtime here does not depend on the size of $\mathcal{H}$. Indeed, the computation is just a series of subsumption checks between the F_i's and $\neg c$, wherein the complexity is only given by the sizes of these clauses.

Of course, exact probability inference is replaced by faster sampling-based approximative methods in the MLN framework. On the other hand, the unification-based algorithm for subsumption checking that we used above can also be replaced by much faster subsumption testers as we have already commented. An experimental comparison including such features is left for an extended version of the paper.

Lastly, the comparison above regarded the computation of the unnormalized quantities $Z_M P_M(c)$ and $Z_S P_S(c)$. Exact computation of $P_S(c)$ would obviously be intractable since it would involve a loop over $3^{|\mathcal{H}|}$ conjunctions. This could be remedied by Monte-Carlo sampling (independent of any randomization possibly adopted in the subsumption-checking step) as in the MLN framework, except that each atom would be in one of 3 (positive, negative, absent) rather than 2 states. Nevertheless, we will now exemplify a situation where we can simply rely on the unnormalized quantity.

4 Discriminative Learning

In the MLN context, discriminative learning usually refers to the situation where the probabilistic model is used to predict the truth value of some atoms given the truth value of other atoms. Here we rather adopt a view more usual in supervised machine learning, where examples (here, ground conjunctions) are partitioned into classes (here we assume exactly two classes) and the model is used to predict the classes of given examples. A straightforward way to classification, based on likelihood odds, is to predict class 1 if $P_S^1(c)/P_S^2(c) > \tau$, where P_S^1 and P_S^2 are two class-specific distributions in the form (4) and τ is a real threshold. The inequality can be rewritten as

$$\frac{Z_S^1 P_S^1(c)}{Z_S^2 P_S^2(c)} > \tau \frac{Z_S^1}{Z_S^2} \equiv \tau' \tag{12}$$

From the training set, we would estimate the structure (F_i's) and parameters (w_i's) for both distributions as well as the parameter τ' with the objective of maximizing training-set accuracy (modulo overfitting counter-measures). As follows from the expression above, we only need the unnormalized quantities $Z_S^1 P_S^1(c)$ and $Z_S^2 P_S^2(c)$ along with the learned parameter τ' to classify c.

Perhaps a more pragmatic way to classification learning, however, is to drop the exponential form while maintaining the proposed $m_i(c)$ concept. We would classify c into class 1 iff $\sum_{i=1}^n w_i m_i(c) > \tau$. This form gives a simple clue for structure learning, in particular, we would search for clauses F_i that contradict as many (few) as possible training examples of class 1 (2). Then, parameters w_i and τ would be tuned. This approach in fact corresponds to a sort of propositionalization with subsequent learning of a linear classifier. The kind of propositionalization entailed by the present framework differs from current propositionalization approaches mainly in that 1) full (function-free) clausal logic is used to express features F_i (in current propositionalization systems, features are usually queries or Horn clauses), and 2) rather than a Boolean value, a feature is assigned an integer for a given example, capturing the number of the feature's groundings that contradict the example.

5 Conclusion

We have proposed satisfiability machines, a conceptual simplification of Markov Logic Networks. A full discussion of its relationships to propositionalization as well as classification experiments on gene expression data under gene-ontology background knowledge is left for an extended version of this paper.

Acknowledgement. This work was supported by the Czech Science Foundation project P103/10/1875 "Learning from Theories".

References

1. De Raedt, L.: Logical settings for concept-learning. Artificial Intelligence 95(1), 187–201 (1997)
2. Kuzelka, O., Zelezny, F.: Fast estimation of first-order clause coverage through randomization and maximum likelihood. In: ICML 2008: 25th Int. Conf. on Machine learning (2008)
3. Kuzelka, O., Zelezny, F.: A restarted strategy for efficient subsumption testing. Fundamenta Informaticae 89(1), 95–109 (2008)
4. Kuzelka, O., Zelezny, F.: Block-wise construction of tree-like relational features with monotone reducibility and redundancy. Machine Learning 83(2), 163–92 (2011)
5. Maloberti, J., Sebag, M.: Fast theta-subsumption with constraint satisfaction algorithms. Machine Learning 55(2), 137–74 (2004)
6. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning 62(1-2) (2006)
7. Santos, J., Muggleton, S.: Subsumer: A Prolog theta-subsumption engine. In: ICLP'2010: Int. Conf. on Logic Programming (2010)
8. Singla, P., Domingos, P.: Markov logic in infinite domains. In: 23rd Conference on Uncertainty in Artificial Intelligence. AUAI Press (2007)