Combining Subjective Probabilities and Data in Training Markov Logic Networks

Tivadar Pápai[1], Shalini Ghosh[2], and Henry Kautz[1]

[1] Department of Computer Science,University of Rochester, Rochester, NY,
{papai,kautz}@cs.rochester.edu,
[2] Computer Science Laboratory, SRI International, Menlo Park, CA,
shalini@csl.sri.com

Abstract. Markov logic is a rich language that allows one to specify a knowledge base as a set of weighted first-order logic formulas, and to define a probability distribution over truth assignments to ground atoms using this knowledge base. Usually, the weight of a formula cannot be related to the probability of the formula without taking into account the weights of the other formulas. In general, this is not an issue, since the weights are learned from training data. However, in many domains (*e*.g. healthcare, dependable systems, *e*tc.), only little or no training data may be available, but one has access to a domain expert whose knowledge is available in the form of subjective probabilities. Within the framework of Bayesian statistics, we present a formalism for using a domain expert's knowledge for weight learning. Our approach defines priors that are different from and more general than previously used Gaussian priors over weights. We show how one can learn weights in an MLN by combining subjective probabilities and training data, without requiring that the domain expert provides consistent knowledge. Additionally, we also provide a formalism for capturing conditional subjective probabilities, which are often easier to obtain and more reliable than non-conditional probabilities. We demonstrate the effectiveness of our approach by extensive experiments in a domain that models failure dependencies in a cyber-physical system. Moreover, we demonstrate the advantages of using our proposed prior over that of using non-zero mean Gaussian priors in a commonly cited social network MLN testbed.

1 Introduction

Markov logic [1], a language widely used for relational learning, represents knowledge by a set of weighted first-order logic formulas. However, except for Markov Logic Networks (MLNs) with special structure, the weights cannot be interpreted as probabilities or simple functions of probabilities. The probability of a particular weighted formula can only be computed by taking into account all of the weights in all of the formulas in the full grounding of the MLN. When weights are learned from training data without any prior knowledge, the non-informative nature of individual weights is not problematic. However, in many domains, one may have little or no training data, but instead have access to a domain expert's *subjective probabilities* and *subjective conditional probabilities*. For example, in the healthcare domain, one might have little data about certain rare diseases, but a doctor may have a subjective notion of what percentage of a rare disease occurs among her patients. Another domain we consider in our experiments models fault-tolerant systems. There may be a paucity of failure data, but an engineer could supply subjective conditional probabilities such as, “If the failure probabilities of the individual components of a system are of the order of 10^{-3}, then the overall system failure probability is of the order of 10^{-4}”. In this paper, we provide a formal account of how such domain knowledge can be incorporated into an MLN. Our approach applies to arbitrary MLNs, and in particular, is not restricted to the known special cases of MLNs whose structure corresponds to Bayesian Networks or *chordal graphs* (which are discussed in more detail below). We describe two approaches for encoding domain knowledge as *priors*: the first requires the expert's knowledge to be a consistent set of non-conditional probabilities, while the second, more general, approach allows inconsistent knowledge and conditional probabilities, but has a non-convex optimization subproblem.

We also demonstrate that earlier approaches to incorporating knowledge by defining non-zero mean Gaussian priors over the weights of a MLN (*e.g.*, as implemented in Alchemy [2]) can only be justified in MLNs with special structure, and even then they have certain disadvantages compared to our approach.

The rest of the paper is organized as follows: Sec. 2 covers the mathematical background; Sec. 3 shows the connection between the expected values of features of MLNs and the subjective probabilities provided by an expert; Sec. 4 discusses the disadvantages of using Gaussian priors on the weights of an MLN; Secs. 5 and 6 define the two different type of priors we investigate; Sec. 7 describes our experiments; and Secs. 8 and 9 discuss related work, summarize our results, and lay out our planned future work.

2 Background

2.1 Markov Logic Network

Markov logic [1] is a knowledge representation language that uses weighted formulas in first-order logic to compactly encode probability distributions over relational domains. A *Markov logic network* is a set of weighted first-order logic formulas and a finite set of constants $C = \{c_1, c_2, \dots, c_{|C|}\}$ which together define a Markov network $M_{L,C}$ which contains a binary node for each possible grounding of each predicate (ground atom) and a binary valued feature for each grounding of each first-order logic formula. In each truth assignment to the ground atoms, the value of a node is 1 if the corresponding ground predicate is *true*, and 0 otherwise. Similarly, the value of a feature is 1 if the corresponding ground formula is *true*, and 0 otherwise. In this paper we assume function-free clauses and Herbrand interpretations. The probability of a truth assignment (world) x to the ground atoms in an MLN is defined as:

$$Pr(X = x|w) = \frac{\exp(\sum_i w_i n_i(x))}{Z(w)}, \tag{1}$$

where $n_i(x)$ is the number of true groundings of the i-th formula, w_i is the weight of the i-th formula and Z is the normalization factor. We sometimes refer to ground atoms as (random) variables, but these are not to be confused with the quantified variables (ranging over C) that appear in first-order formulas.

2.2 Exponential Families of Probability Distributions

The probability distributions defined by Markov Logic Networks belong to the exponential families of distributions [3], since (1) can be rewritten in a more general form:

$$\Pr(X = x) = \exp\left(\langle \theta, f(x) \rangle - A(\theta)\right) \ , \tag{2}$$

where θ_i are the *natural parameters* of the distribution, f_i are the features, and $A(\theta)$ is responsible for the normalization. As one can tell by comparing (1) and (2), θ corresponds to w, f_i to n_i and $A(\theta) = \log Z(w)$. The probability (likelihood) of training data $\mathcal{D} = \{x_1, \ldots, x_N\}$ is (with the usual *i.i.d.* assumption):

$$\Pr(\mathcal{D}) = \exp\left(\left\langle \theta, \sum_{d=1}^{N} f(x_d) \right\rangle - N \cdot A(\theta)\right) \ , \tag{3}$$

As we can see, (3) depends upon the data only through $\sum_{d=1}^{N} f(x_d)$ (a *sufficient statistic* of the data set $\mathcal{D}$). Let $f(\mathcal{D}) = \sum_{d=1}^{N} f(x_d)$ and $\overline{f}(\mathcal{D}) = \frac{1}{N}\sum_{d=1}^{N} f(x_d)$. θ is usually set to maximize (3) for the given training data.

In (2) the distribution is parameterized by its natural parameters (θ). However, it can also be parameterized by its *mean parameters*, where the means are defined as the expected values of the features:

$$\mu_i = \sum_{x} f_i(x) Pr(X = x) = \mathbb{E}\left[f_i\right] \ . \tag{4}$$

There is a many-to-one mapping from θ to μ. We use $\theta_{F(x)}$ and $\mu_{F(x)}$ to denote the component of the vectors corresponding to the feature representing the true groundings of formula $F(x)$, and in general follow this convention for vectors of parameters. We will use the notation $\mu(\theta)$ when we want to emphasize the dependence of μ on θ. Since either of μ or θ completely determine the distribution, we have the choice of defining a prior either over μ or θ to represent the knowledge of the expert. The prior we will define over θ restricts the kind of subjective probabilities the expert provides, but will result in a convex optimization problem, thereby making the approach computationally attractive. On the other hand, the prior we will define over μ is less restrictive, allowing both conditional and inconsistent probabilistic constraints, at the worst-case cost of requiring the solution of a non-convex optimization problem. However, we will also discuss special cases when the optimization problem can be solved by simple gradient descent, or at least guarantees can be given for the quality of the result found at any point where the gradient becomes zero.

3 Relationship Between Subjective Probabilities and the Parameters of the Exponential Family

We consider the case where a domain expert provides subjective probabilities for some or all of the formulas in an MLN. For example, if $F(x)$ is a formula where x is a vector of (implicitly) universally-quantified variables, the expert can estimate how likely it is that a randomly chosen grounding of $F(x)$ is true. The expert may also provide subjective conditional probabilities over ground formulas. For example, if $F_1(x_1)$ and $F_2(x_2)$ are formulas, then for chosen groundings c_1 and c_2, where c_2 contains constants only from c_1, the expert may estimate the probability that if $F_1(c_1)$ is true then $F_1(c_1) \wedge F_2(c_2)$ will be true as well. We will denote the former statistic by $S\mathrm{Pr}(F(x))$ and the latter by $\mathrm{SPr}(F_2(c_2)|F_1(c_1))$. For example, $\mathrm{SPr}\left(\mathrm{Cancer}(c)|\mathrm{Smokes}(c)\right) = 0.4$ means that if the chosen individual c smokes, (s)he has lung cancer as well with probability 0.4 according to the expert. Similarly, $\mathrm{SPr}\left(\mathrm{Smokes}(X)\right) = 0.01$ states the percentage of the population that smokes in the opinion of the expert. If the MLN happens to be *symmetric* in the sense that for any bindings of x_1 and x_2 to constant vectors c_1 and c_2, $\mathrm{SPr}(F_2(c_2)|F_1(c_1))$ is constant, we allow the notation $\mathrm{SPr}(F_2(x_2)|F_1(x_1))$ where x_2 only contains variables from x_1. *W.l.o.g.* we henceforth assume that $x_1 = x_2$ in any subjective probabilities.[3]

Let $g(F(x))$ denote the total number of groundings of formula $F(x)$ and let $\overline{\mu}_{F(x)} = \frac{\mu_{F(x)}}{g(F(x))}$. Given the definition of $\mathrm{SPr}(F_2(x)|F_1(x))$ and $\mathrm{SPr}(F(x))$, an initial idea would be to take $\overline{\mu}_{F(x)}$, $\overline{\mu}_{F_2(x)\wedge F_1(x)}$, and $\overline{\mu}_{F_1(x)}$, and try to satisfy $\overline{\mu}_{F(x)} = \mathrm{SPr}(F(x))$ and $\mathrm{SPr}(F_2(x)|F_1(x)) = \frac{\overline{\mu}_{F_2(x)\wedge F_1(x)}}{\overline{\mu}_{F_1(x))}}$ for every given subjective (conditional) probability, in absence of training data. In many cases, however, it is impossible to match the subjective probabilities of the expert. For example, consider the case where according to the expert $\mathrm{SPr}(P(x)) = 0.5$ and $\mathrm{SPr}(P(x) \vee Q(x)) = 0.4$. It is easy to see that in this situation no vector θ would provide a normalized μ that would match both subjective probabilities. We will call a set S of subjective (conditional) probabilities *inconsistent* in an MLN M that has all the formulas occuring in S if there does not exist any θ such that $\overline{\mu}_{F(x)}(\theta) = \mathrm{SPr}(F(x))$ and $\frac{\overline{\mu}_{F_2(x)\wedge F_1(x)}(\theta)}{\overline{\mu}_{F_1(x)}(\theta)} = \mathrm{SPr}(F_2(x)|F_1(x))$ for every $\mathrm{SPr}(F(x)), \mathrm{SPr}(F_2(x)|F_1(x)) \in S$. It can be proven that S is inconsistent in M if and only if there is not any distribution that would satisfy all the probabilistic constraints in S.

[3] Fisseler [4] explains in more details why conditional probability constraints must be dealt with at the ground level in Markov Logic-like relational probabilistic logics in order to match our definition for conditional probability. Thimm *et. al* [5] provide several different semantics for defining first-order conditional probabilities in probabilistic logics. The symmetric case described above corresponds to what they call *aggregating semantics*. It is beyond the scope of our paper to examine all the ways in which first-order conditional probabilities could be defined. For the sake of simplicity in rest of the paper we assume that subjective conditional probabilities are either defined at the ground level or are symmetric (*i.e.*, use aggregating semantics).

We will call a set of subjective (conditional) probabilities *fully specified* if for every subjective conditional probability $\mathrm{SPr}(F_2(x)|F_1(x))$, a value for $\mathrm{SPr}(F_1(x))$ is provided by the expert as well. In the case of fully specified subjective probabilities, we can replace a constaint involving $\mathrm{SPr}(F_2(x)|F_1(x))$ by the constraints $\overline{\mu}_{F_1(x)} = \mathrm{SPr}(F_1(x))$ and $\overline{\mu}_{F_2(x)\wedge F_1(x)} = \mathrm{SPr}(F_2(x) \wedge F_1(x))$.

In Sec. 5, we define a prior over θ assuming that the domain expert provides a consistent and fully specified set of subjective probabilities. This is a realistic assumption if the expert's subjective probabilities are not literally subjective, but have foundations in the statistics of real world data (*e.g.* 20% of US adults smoke). In Sec. 6, we allow inconsistent subjective conditional and non-conditional probabilities, with the tradeoff of possibly requiring greater computational effort.

4 Gaussian Priors and Chordal Graphs

Before describing our proposed solution for incorporating an expert's knowledge into an MLN, we discuss the idea of using non-zero mean Gaussian (or Laplace) priors to represent preference for subjective probability values [6]. We will demonstrate that using log-odds or log-probabilities as means of Gaussian (or Laplace) priors can be used under special circumstances. However, the standard deviation of each Gaussian needs to be scaled based on the associated probability of the formula, and rewriting an arbitrary MLN to put it into the required form may cause an exponential increase in size. Our examples require only the propositional subset of Markov Logic.

The Alchemy Tutorial [2] describes how one can convert a Bayesian Network into a propositional Markov Logic knowledge base. In the conversion, each entry in the conditional or non-conditional probability table for a node generates a clause whose weight is the negative log of the probability. In the problem at hand, however, we *begin* with an MLN, not a Bayesian Network.

Chordal graphs are the subset of undirected graphical models which correspond to both directed and undirected graphical models. We show that the problem of representing consistent expert knowledge in an MLN whose underlying Markov Random Field is chordal can be solved efficiently. Suppose we have a propositional knowledge base to which the corresponding ground MRF is a chordal graph G. It follows that the probability model P (represented by the ground Markov Network) is decomposable [7, 8], *i.e.*, the joint probability of its random variables can be represented as the product of the joint probabilities of the variables in the individual cliques divided by the product of the joint probabilities of random variables in certain intersections of certain pairs of cliques. More precisely, let $C_1, \ldots, C_n$ be the sets of variables belonging to each maximal clique ordered by a maximum cardinality ordering, and let $C_{j(i)}$ be the unique predecessor of C_i in a join tree corresponding to G. Then the joint probability can be expressed as, $\Pr(\cup C_i = x) = \frac{\prod_i \Pr(C_i = c_i)}{\prod_i \Pr(C_i \cap C_{j(i)} = c_i \cap c_{j(i)})}$. For a clique C with variables $X_1, \ldots, X_n$ we will call a set S_C of conjunctions a *cover* of C if S_c contains all the possible 2^n different conjunctions over $X_1, \ldots, X_n$. To be able to use the log probabilities, we require that the formulas present in the knowledge base contain exactly the conjunctions that cover every C_i clique and $C_i \cap C_{j(i)}$ intersection of cliques.

Assume now that there is a domain expert who specifies consistent probabilities for all the formulas in the knowledge base. Let T be a truth assignment to all the variables in the MLN. Let $t_{C=T}$ be a conjunction corresponding to the truth assignment in clique C (or intersection of cliques) that agrees with T, and let $p_{t_{C=T}}$ denote its probability (this probability can be unknown, *i.e.*, needed to be learned or specified by the expert). In this setting, the probability of a truth assignment T to all the variables can be written as:

$$\begin{aligned}
\Pr(T) = \frac{\prod_i p_{t_{C_i=T}}}{\prod_i p_{t_{C_i \cap C_{j(i)}=T}}} &= \exp(\sum_i \ln p_{t_{C_i=T}} - \sum_i \ln p_{t_{C_i \cap C_{j(i)}=T}}) \qquad (5)\\
&= \exp\Bigg(\sum_i \sum_{T' \in \mathcal{T}_{C_i}} \ln p_{t_{C_i=T'}} f_{t_{C=T'}}(T) \\
&- \sum_i \sum_{T' \in \mathcal{T}_{C_i \cap C_{j(i)}}} \ln p_{t_{C_i \cap C_{j(i)}=T'}} f_{t_{C_i \cap C_{j(i)}=T'}}(T) \Bigg) ,
\end{aligned}$$

where $\mathcal{T}_C$ is the set of all truth assignments over the variables in clique C and $f_{t_{C=T'}}$ corresponds to the feature which represents the conjunction belonging to $t_{C=T'}$, *i.e.*, 1 if the conjunction is true, otherwise it is false. It is easy to see that the last line in (5) corresponds to a Markov Logic representation and that for every truth assignment T the MLN gives back the correct probability, with no normalization needed. Thus, chordal MLNs have the advantage that one can use log-probabilities as priors on the weights. However, MLNs are not, in general, chordal, and modifying an MLN to make it chordal — *i.e.* adding formulas that triangulate the underlying graph — can increase its size exponentially [9].

A second disadvantage of the approach just described is that Gaussian (or Laplace) priors on the natural parameters do not translate to Gaussian (or Laplace) priors in the mean parameter space — that is, in the space of probabilities as opposed to weights. Intuitively, one would want to use the variance of the prior to control how close the parameter is to the subjective probability after training. However, distance in the θ space does not linearly transform to a distance in the μ space. *E.g*, consider having one formula $F(x)$. A change of its weight from 0 to 1 would change its probability from 0.5 to ≈ 0.73, while a change from 1000 to 1001 would practically not change its probability. At a minimum, we would need to scale the standard deviations of the Gaussian priors according to the mean parameter of the distribution. This distinction is not present in (3).

Moreover, in many cases the expert can only know the (subjective) probabilities of a subset of formulas. Even if this subset of the formulas spans a chordal MLN satisfying the conditions to use log-probabilities, we still cannot use log-probabilities as weights if with the rest of the formulas altogether we do not have a chordal MLN. To illustrate this problem, consider the case when we have $N+1$ ground atoms $p, q_1, \ldots, q_N$ in the domain and the expert knows exactly the probability of p being true, or equivalently provides the value of $o = \frac{\mathrm{SPr}(p)}{1-\mathrm{SPr}(p)}$. Further assume we also have formulas $p \vee q_i$ in our knowledge base for every $i = 1, .., N$, and learn the weights of these formulas from a training data set for which we know $\mu_p = \mathrm{SPr}(p)$ holds. If we use a strong non-zero mean Gaussian prior over the weight of p centered around $\log o$, *i.e.*, we fix the weight of p during the weight learning, then if w_i is the weight needed for $p \vee q_i$ to match the empirical feature count from the data, we will get $\frac{\mathrm{Pr}(p)}{1-\mathrm{Pr}(p)} = \frac{\mathrm{SPr}(p) \sum_{q_1,\ldots,q_N} \prod_{i=1}^{N} \exp(w_i)}{(1-\mathrm{SPr}(p)) \sum_{q_1,\ldots,q_N} \prod_{i=1}^{N} I[q_i] \exp(w_i)} = o \prod_{i=1}^{N} \frac{2\exp(w_i)}{\exp(w_i)+1}$, where $I[q_i]$ is the feature corresponding to $p \vee q_i$ with the substitution $p = \text{false}$, *i.e.*, $I[q_i] = 1$ if q_i is *true*, otherwise 0. (We do not have a feature in the numerator, since $p \vee q_i$ is always true when $p = \text{true}$.) It is easy to see that as we increase N our wrong choice of prior can cause an arbitrarily large deviation in the learned marginal of p from the correct one (consider the case when μ_p is close to 0, so using the expert's probability we set *e.g.*, $w_p = -100$ while $\mu_{p \vee q_i}$ is close to 1, and *e.g.* after the weight learning we have $w_i = 200$ for every i). We will demonstrate this downside of the prior further in Sec. 7 with experiments.

5 Defining a Prior on the Natural Parameters

In this section we consider the case when the expert gives us a fully specified set of consistent subjective probabilities, and show that we can avoid the disadvantages of Gaussian priors defined on the natural parameters. Exponential families of probability distributions have the advantageous property of easily defined conjugate priors [10, 11] —

$$\mathrm{Pr}(\theta|\beta, \alpha) \propto \exp\left(\langle \theta, \alpha\beta \rangle - \alpha A(\theta)\right) \tag{6}$$

— and these priors have an intuitive meaning. β can be related to $\overline{f}(\mathcal{D})$ and α to N if we compare (6) to (3). β takes the role of the average empirical feature count, while α is going to be the pseudo-count. However, if β is not in the set of consistent mean parameters, then (6) is not going to form a distribution (*i.e.*, it is an *improper* prior).

Let $\theta_{F(x)}, \beta_{F(x)}$, and $\alpha_{F(x)}$ denote the components of the vectors which correspond to the feature $f_{F(x)}$. We can capture consistent subjective probabilities by setting $\beta_{F(x)} = \mathrm{SPr}(F(x)) g(F(x))$. This intuitively makes sense, because the knowledge of the expert will be represented as pseudo-data, where α describes the confidence of the expert in his subjective probabilities, and β represent the sufficient statistic belonging to the pseudo-data. The posterior becomes:

$$\mathrm{Pr}(\theta|\mathcal{D}, \beta, \alpha) \propto \exp\left(\left\langle \theta, \left(\alpha\beta + N\overline{f}(\mathcal{D})\right)\right\rangle - (\alpha + N)\, A(\theta)\right) \ . \tag{7}$$

The gradient of the logarithm of the posterior *w.r.t.* θ becomes:

$$\frac{\partial \log \mathrm{Pr}(\theta|\mathcal{D}, \beta, \alpha)}{\partial \theta} = \alpha\beta + N\overline{f}(\mathcal{D}) - (\alpha + N)\mathbb{E}_\theta[f] \ . \tag{8}$$

This shows that the probability of the data and subjective probabilities of the expert are measured on the same scale; thus, we do not have to make adjustments for the subjective probabilities depending on their values, in contrast to the case for Gaussian priors.

The posterior is a log-concave function, which can be verified by taking the derivative of (8) *w.r.t.* θ. Thus, we are guaranteed to find a global maximum of (8) by using a gradient ascent.

This formulation assumes that the subjective probabilities of the expert and the data are coming from domains with the same size, which is not a realistic assumption in statistical relational learning. For example, a doctor can base his estimates on thousands of previously seen patients, while the training data can contain only the data for a small control group. We can allow the domain size to vary by using a distinct A for the data and for the expert in (6), (7), and (8) which would result in the computation of $\mathbb{E}[f]$ w.r.t. two different domain size of MLN but with the same knowledge base. This approach can be generalized to allow the different training examples to come from different sized domains, and to incorporate the knowledge of different experts. Although these modifications are straightforward, the effect of weak transfer [12] may change the meaning of subjective probabilities in different domains. For the sake of simplicity in the rest of the paper we will only consider the base case, *i.e.*, where the training data and the expert's knowledge come from the same domains.

In summary: the approach just described allows us to incorporate consistent prior knowledge by defining a prior over the natural parameters, avoids the problems of Gaussian priors (*i.e.* requiring a chordal structure and difficulty in defining the variance), and requires (only) solving a convex optimization problem. However, inconsistent subjective probabilities are difficult to handle in this approach. Furthermore, the expert is required to specify a subjective probability for every formula.A possible approach that could solve both issues would be to try to define a distribution (a hyper-prior) over the parameters (hyper-parameters) of the prior distribution. However, then for any reasonable choice of hyper-prior we would have the problem of dealing with a normalization factor in (6) that depends on the value of the hyper-parameters. Instead of going this route, in the next section we will define a prior over the mean parameters.

6 Defining Prior on the Mean Parameters

As we pointed out in the last section, defining a prior on θ only works when the subjective probabilities are consistent and the conditional probabilities are fully specified. To overcome these limitations, we soften the constraints by introducing a prior on the set of consistent μ values. Let $\mathit{\Pi}(\mu)$ be the prior on μ. Let $\Pr(\mathcal{D}|\theta) = \prod_{i=1}^{N} \Pr(x_i|\theta)$ be the probability of the i.i.d. training data given θ or μ. The posterior $\Pr(\mathcal{D}|\mu)$ is proportional to $\Pr(\mathcal{D}, \mu)$, hence it is sufficient to maximize $\Pr(\mathcal{D}, \mu)$. The log-probability of the data with the prior can be written as:

$$L(\mathcal{D}, \mu(\theta)) = \ln \Pr(\mathcal{D}|\mu) + \ln \mathit{\Pi}(\mu) \ . \tag{9}$$

We are looking for the weight vector (θ) that maximizes $L(\mathcal{D}, \mu(\theta))$. The gradient of L *w.r.t.* θ is:

$$\frac{\partial L}{\partial \theta} = \frac{\partial \ln \Pr(\mathcal{D}|\theta)}{\partial \theta} + \frac{\partial \ln \mathit{\Pi}(\mu)}{\partial \mu}\frac{\partial \mu}{\partial \theta} = \sum_i \left(f(d_i) - \mu\right) + \mathit{\Sigma}_\theta \frac{\partial \ln \mathit{\Pi}(\mu)}{\partial \mu}\ , \tag{10}$$

where we use the fact that $\frac{\partial \mu(\theta)}{\partial \theta} = \mathit{\Sigma}_\theta$, the covariance matrix of f *w.r.t.* the distribution defined by θ. The concaveness of (9) depends on the choice of $\ln \mathit{\Pi}(\mu)$; in general, $\ln \mathit{\Pi}(\mu)$ may be non-concave. Nonetheless, with a careful choice of $\mathit{\Pi}(\mu)$ there are cases when finding a global optimum of L can be reduced to a convex-optimization problem, or at least a solution with quality guarantees can be found by a gradient ascent algorithm.

6.1 Choosing the Prior

Our goal is that when no training data is available, we would like $\overline{\mu}_{F(x)}$ to be as close to $\mathrm{SPr}(F(x))$ as possible. Similarly, for conditional probabilities we want to match $\frac{\overline{\mu}_{F_2(x) \wedge F_1(x)}}{\overline{\mu}_{F_1(x))}}$ to $\mathrm{SPr}(F_2(x)|F_1(x))$. A prior that can capture this goal is, *e.g.*, a truncated Gaussian distribution over $\overline{\mu}_{F(x)}$, $\frac{\overline{\mu}_{F_2(x) \wedge F_1(x)}}{\overline{\mu}_{F_1(x))}}$ centered around $\mathrm{SPr}(F(x))$ and $\mathrm{SPr}(F_2(x)|F_1(x))$, respectively. We have to truncate the Gaussian since $\overline{\mu}$ is constrained to be between 0 and 1; moreover, the distribution has to be defined over consistent $\overline{\mu}$ values. Since $\overline{\mu}_{F_1(x))}$ can get close to 0, for numerical stability it is more beneficial to try to match $\overline{\mu}_{F_2(x) \wedge F_1(x)}$ to $\overline{\mu}_{F_1(x)}\mathrm{SPr}(F_2(x)|F_1(x))$. The distribution over μ is just a linear transformation of the truncated Gaussian distribution over $\overline{\mu}$ (since the Jacobian $\frac{\partial \overline{\mu}}{\partial \mu}$ is constant), resulting in:

$$\begin{aligned}\Pr(\mu) \propto \exp \Bigg(&- \sum_{F(x)} \alpha_{F(x)} \left(\overline{\mu}_{F(x)} - \mathrm{SPr}\left(F(x)\right)\right)^2 - \\ &\sum_{F_2(x)|F_1(x)} \alpha_{F_2(x)|F_1(x)} \left(\overline{\mu}_{F_2(x) \wedge F_1(x)} - \overline{\mu}_{F_1(x)} \mathrm{SPr}\left(F_2(x)|F_1(x)\right)\right)^2 \Bigg)\end{aligned} \tag{11}$$

In (11), the different α values correspond to the confidence of the expert in his different subjective probabilities*e.g.* $\alpha = 0$ tells that the expert has no information about the subjective (conditional) probability of that feature.

6.2 Cases Solvable by Gradient Ascent

As we mentioned before, there are special cases when we end up with a convex optimization problem or can give guarantees about the quality of the solution of a gradient ascent algorithm. An example for the former is when there is no training data available, the subjective probabilities are consistent, and there are no subjective conditional probabilities given, since then $\mathit{\Pi}(\mu)$ takes its maximum when the exponent in (11) is 0. Consequently, we can just find the point where

$\forall F(x) : \overline{\mu}_{F(x)} - \mathrm{SPr}\left(F(x)\right) = 0$ using regular gradient ascent (without the presence of $\varSigma_\theta$ in (10)). (This is exactly equivalent to using the prior over θ defined in (6) where the same $\alpha_{F(x)}$ is used for every formula.)

Another important case is when we do not have a convex-optimization problem, but can still guarantee that all the points where the gradient is 0 are located close to a global optimum, uses a Gaussian prior for $\varPi(\mu)$. More precisely:

Proposition 1 *The gradient ascent algorithm always converges to a stationary point θ, where $\overline{\mu}(\theta)$ is guaranteed to fall within an $\epsilon(N)$ radius of $\hat{f}(\mathcal{D})$ if $\forall i$: $0 < \hat{f}_i(\mathcal{D}) < 1$, where $\hat{f}_{F(x)}(\mathcal{D}) = \frac{\overline{f}_{F(x)}(\mathcal{D})}{g(F(x))}$, N is the amount of training data and $\epsilon(N)$ is a strictly monotonically decreasing function of N and $\lim_{N\to\infty} \epsilon(N) = 0$.*

Proof. (Sketch) We start with showing that for all the θ's where this gradient is 0, it is true that $\overline{\mu}(\theta)$ has to be within $\epsilon(N)$ radius of $\hat{f}(\mathcal{D})$. $\varSigma_\theta$ is a covariance matrix of bounded valued random variables, hence the entries in $\varSigma_\theta$ are bounded as well. Also, both $\mathrm{SPr}(F(x)) - \overline{\mu}_{F(x)}$ and $\overline{\mu}_{F_2(x)\wedge F_1(x)} - \overline{\mu}_{F_1(x)}\mathrm{SPr}(F_2(x)|F_1(x)$ are bounded; consequently, the vector $g(\theta) = \varSigma_\theta \frac{\partial \ln \varPi(\mu)}{\partial \mu}$ is bounded, and there is a bound that does not depend on N or θ. Let the components of vector b be the tightest bounds on the absolute values of the components of g, *i.e.*, $b_i = \sup_\theta\{|g_i(\theta)|\}$. Let $g'(\theta) = f(\mathcal{D}) - N\mu(\theta) = N(\overline{f}(\mathcal{D}) - \mu(\theta))$. In (10) the gradient can only be 0 if $g' + g = 0$. A necessary condition for this $|g_i'(\theta)| \leq b_i$ for every $1 \leq i \leq n$, equivalently $|\overline{f}_i(\mathcal{D}) - \mu_i(\theta)| \leq \frac{b_i}{N}$, *i.e.*, $|\hat{f}_i(\mathcal{D}) - \overline{\mu}_i(\theta)| \leq \frac{b_i}{g_i(x)N} = \epsilon(N)$. Furthermore, the normalized gradient is directed towards the sphere centered around $\hat{f}(\mathcal{D})$ with radius $\epsilon(N)$. This completes the proof.

Since $\overline{f}(\mathcal{D})$ is the value to which μ would converge in absence of subjective probabilities, this result ensures that with sufficient training data μ converges to the same value it would converge to without having the prior, which is a desideratum for any prior. This results always holds as long as the gradient of $\ln \varPi(\mu)$ is bounded. The analogue of Proposition 1 holds, when we increase the domain size, instead of the number of training data sets. (*I.e.*, if the expert bases his statistics on seeing a population of 1000 people, the weight of his knowledge becomes negligible when we have a training data set with, *e.g.*, 1000000 people.)

Thus, despite that L can have more than one local optimum, a gradient ascent algorithm will serve as a basis for our optimization algorithm. We will empirically demonstrate that even when the conditions of the (approximation) guarantee do not hold, we can still be better off incorporating subjective probabilities rather than solely relying on the available data.

7 Experiments

Our implementation is an extension to the Probabilistic Consistency Engine (PCE) [13], an open source MLN inference tool that has been used effectively in different problem domains[13, 14]. We used stochastic gradient ascent with a monotonically decreasing learning rate to find a point where the gradient is 0, and added a regularizer so that in the guaranteed convex cases we would always end up with a unique solution. We also experimented with taking random steps occasionally to avoid becoming stuck in a local optimum and to increase our chance of reaching the global optimum. To compare the quality of the weight vectors that our weight learning algorithm visited, we would need to evaluate the log-posteriors at the visited points. However, since computing the real log-posteriors in (9) is costly, we use the combination of the pseudo log-likelihood of the data along with the log-probability of the prior for the comparison, to select the best weights. In our experiments, we always ended up using weight vectors where the gradient ascent algorithm with decreasing learning rate stopped, thus, simple gradient ascent provided satisfactory results in our examples. Note that, this can be due to the fact that some random noise is always present in the sampling algorithm, depending on how many samples we are using to approximate μ in (10) which helped get out from the local optima.

The goal of our experiments is to demonstrate the benefits of our model and show the advantages of using our proposed prior compared to the one currently available in Alchemy, namely Gaussian non-zero mean priors. In our first batch of experiments, we used an MLN that models the failure dependencies in a cyber-physical system, the Cabin Air Compressor (CAC)[15]. The CAC is an electromechanical subsystem in an Aircraft Environmental Control Systems (ECS), which is used to deliver fresh air to the cabin and provide pressurization. The MLN models a voting-3-CAC architecture, where three CAC subsystems are connected in a voting configuration, such that the overall system fails only if at least two out of the three CAC subsystems fail. A high load is put on a CAC if one or more of the other two CACs fail, and putting high load on a CAC increases its probability of failure. The MLN models the failure probability of the overall system, given the failure probabilities of each subsystem, and taking into account the effects of high load interactions. The voting-3-CAC MLN we used in our experiment had 4 hard and 4 soft clauses. We abbreviate predicates $failCac$, $failSystem$, $failCacHighLoad$ with C, S and H respectively. The hard and soft clauses we had in our KB are in Table 1. We resorted to the use of synthetic data, because in the real world system the probability of system failure is so low that acquiring real world data set that could capture the underlying distribution would require a lot of time. We hand-tuned the weights for the soft

<table>
<tr><th colspan="2">Hard Clauses</th></tr>
<tr><td colspan="2">$\forall c, d, e, s : C(c) \wedge C(d) \wedge C(e) \wedge (c \neq d) \wedge (d \neq e) \wedge (c \neq e) \supset S(s)$</td></tr>
<tr><td colspan="2">$\forall c, d, e, s : C(c) \wedge C(d) \wedge \neg C(e) \wedge (c \neq d) \wedge (d \neq e) \wedge (c \neq e) \supset S(s)$</td></tr>
<tr><td colspan="2">$\forall c, d, e, s : C(c) \wedge \neg C(d) \wedge \neg C(e) \wedge H(d) \wedge H(e) \wedge (c \neq d) \wedge (d \neq e) \wedge (c \neq e) \supset S(s)$</td></tr>
<tr><td colspan="2">$\forall c, d, e, s : C(c) \wedge \neg C(d) \wedge \neg C(e) \wedge H(d) \wedge \neg H(e) \wedge (c \neq d) \wedge (d \neq e) \wedge (c \neq e) \supset S(s)$</td></tr>
<tr><th colspan="2">Soft Clauses</th></tr>
<tr><td>-1.03594</td><td>$\forall c, d : failCac(c) \wedge \neg failCac(d) \wedge c \neq$d</td></tr>
<tr><td>-0.9857</td><td>$\forall c, d : failCacHighload(d) \wedge failCac(c) \wedge \neg failCac(d) \wedge c \neq d$</td></tr>
<tr><td>-0.491191</td><td>$\forall c : failCac(c)$</td></tr>
<tr><td>-1.01143</td><td>$\forall s : failSystem(s)$</td></tr>
</table>

Table 1: Hard clauses and soft clauses with their weights

clauses to get a realistic model. We generated 10 synthetic training data sets each with 10, 20, 50, 100, 200, 500, 1000 and 10000 samples by collecting samples from the MLN using MC-SAT [16], and then selected only a subset of the samples in order to increase the independence between them. The normalized expected feature counts $\overline{\mu}$ varied between 0.01 to 0.3 for the soft clauses. We added a random value from $\{-\delta\overline{\mu}, +\delta\overline{\mu}\}$ to every normalized feature count to represent the uncertainty of the expert in his beliefs. We varied $\delta = 0.0, 0.1, 0.2, 0.3$ in our 4 different noise models (noiseless, noisy 1-3). In our experiments we set 3 subjective probabilities and 1 subjective conditional probability according to these sampled values (cases $noisy1$,$noisy2$ and $noisy3$ in Figure 1). Also, we used subjective probabilities computed from $\overline{\mu}$ for the noiseless case. We computed the KL-divergences $D_{KL}(P\|Q)$ between every learned distribution Q and the real distribution P, and averaged them for the 10 different sample sets. In the noiseless case, we used the log-posterior with the gradient specified in (8) (fully specified set of subjective probabilities). In the noisy cases, we used gradient ascent according to the gradient in (10) using the prior in (11). We set the weight of our prior to match approximately 100 samples from the training data. As we see from the figure, the KL-divergence is relatively high without using our prior for 10 samples, and around 500 samples starts converging to the same values in all cases. This can be explained by the fact that for certain formulas, $\overline{\mu}$ has low values – so consequently for a few samples the formulas corresponding to the low expected average feature counts are unlikely to be satisfied in any of the generated samples.The KL-divergence is finite in these cases due to the use of a regularizer, which prevents the weights from becoming infinite. Arguably other measures ($e.g.$ L_1-norm) could be used instead of KL-divergence. However, in case of a fault tolerant system, one has to consider the associated penalty when a system is claimed to be fail-safe, but in reality has a non-zero probability for failure – L_1 distance measure would not capture this.

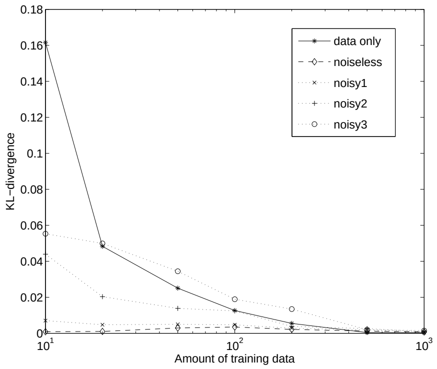

Fig. 1: The averaged KL-divergences measured from the true distribution for the 4 noise models

In summary, when small amount of training data is used, it is beneficial to train the MLN using the subjective probabilities of a domain expert, even if he is not completely confident in his subjective probabilities.

In our second set of experiments, we used a modified version of the “smoking” social network example MLN from the Alchemy Tutorial to analyze the problems of using log-probabilities as weights. We created two different knowledge bases A and B for our experiments. Both knowledge bases used the formulas from Table 2; knowledge base A used 1-5 while B from 1-7. For each knowledge base we ran two sets of experiments, one with using our prior in the mean parameter space, and one with using Gaussians in the natural parameter space centered around the logarithm of the value of the appropriate (conditional) probabilities. The goal of the experiments were to use a strong prior (with small variance and high α, respectively for the non-zero mean Gaussian over the weights and our truncated Gaussian prior,) forcing the formulas $1-5$ which only appear in KB A to have log-probability weight/subjective probability to be equal to the values provided by the expert. About the probabilities of formulas $6-7$ the expert had no information, hence their weights could vary freely in both cases during the weight learning. We had 8 people in the domain. We considered 3 sets of weights and generated 100 samples from the distribution represented by the MLN that had all the formulas in the table. We again created our training data sets by using samples from MC-SAT. We computed the feature counts from the samples and, after normalizing them, we set the (log) probabilities and conditional probabilities of the appropriate formulas in Table 2. In the experiments which used KB A, both priors performed similarly. However, using log-probability weights for formulas 1-5 in KB B proved to be a bad estimate – this is because after weight learning, the formulas 6-7 had non-zero weights, thereby changing the probabilities of formulas 1-5 whose weights were fixed using the desired values. Because the domain was too large to compute the KL-divergence, we measured the L_1-distance between the normalized value of the expected feature counts (probabilities) we get for formulas 1-5, while using the same priors in KB A and B. The normalized feature counts of formulas 1-7 in the 3 experiments and the measured L_1-distances are in Tables 3 and 4. These experiments confirmed that using log-probability weights as means of Gaussian priors can decrease the quality of the learned distribution, even if the expert has access to the true probability values for a subset of the formulas, and motivates the use of our proposed prior.

Formula No.	Formula
1	$\forall x : Smokes(x)$
2	$\forall x : Smokes(x) \wedge Cancer(x)$
3	$\forall x : Smokes(x) \wedge \neg Cancer(x)$
4	$\forall x : \neg Smokes(x) \wedge Cancer(x)$
5	$\forall x : \neg Smokes(x) \wedge \neg Cancer(x)$
6	$\forall x, y : Friends(x, y) \wedge Smokes(x) \supset Smokes(y)$
7	$\forall x, y : Relatives(x, y) \wedge Cancer(x) \supset Cancer(y)$

Table 2: Knowledge bases for the different experiments

Exp.no.	Probabilities of Formulas						
	1	2	3	4	5	6	7
1	0.6058375	0.4609375	0.144825	0.00741250000002	0.38695	0.803438	0.973437
2	0.7480875	0.711075	0.0367375	0.0083125	0.2441	0.862812	0.963750
3	0.68445	0.6442875	0.0404375	0.0749499999999	0.2404625	0.839688	0.943125

Table 3: The probabilities of the formulas in the different experiments.

Experiment	L_1 distance	
	our proposed prior	log-probability weights
1	< 0.05	0.524025
2	< 0.05	0.2613875
3	< 0.05	0.2862375

Table 4: The L_1 distances between the normalized expected feature counts and the empirical feature counts, using the two different priors.

8 Related Work

How the knowledge of an expert can be represented as parameter constraints for Bayesian Networks is discussed in [17, 18]. Although their formalisms can incorporate complex knowledge, it is still restricted to Bayesian networks. Prior probability distributions within the framework of Bayesian statistics in Markov Random Fields have been used successfully in many domains, such as computer vision [19, 20]. For exponential families other methods have been proposed to incorporate information based on expectations of features *e.g.* in [21] or *measurements* in [22], but their models are not tailored to MLNs and *e.g.*, do not allow directly constraining the ratio of expectations of features which is needed to capture conditional probabilities of formulas. The closest relevant research on representing knowledge given as (conditional) probabilities of formulas in MLNs is the work described in [4]. There, the language of Markov logic is extended by probabilistic constraints, which can either be probabilities of universally quantified formulas or conditional probabilities of propositional formulas. However, their framework only handles consistent constraints and does not allow combining the prior knowledge with training data. Eliciting probabilities from expert opinion has been used in research in other areas. For example, [23] shows how to extend de Finettis betting-odds method for considering subjective beliefs regarding ambiguous events.

9 Conclusion

In this paper, we presented a mathematical framework for incorporating subjective probabilities into a Markov Logic Network within the framework of Bayesian statistics. We discussed the benefits and limitations of defining the prior over the natural parameters (weights) versus the mean parameters (probabilities) of the MLN, and demonstrated that earlier approaches based on Gaussian priors over the natural parameters were inadequate. Our framework allows knowledge about conditional subjective probabilities to be incorporated as well as non-conditional probabilities, and can extended to priors where subjective probabilities come from multiple experts. When we are provided with a fully specified and consistent set of subjective probabilities, defining the prior on the natural parameters results in a convex-optimization problem, which is an advantage in terms of computational effort. It is often the case, however, that domain expert subjective probabilities are not consistent. We showed how to make use of possibly inconsistent subjective probabilities by defining a prior over the mean parameters. This approach allows for more flexibility, but at possibly greater computational cost for learning, because the optimization problem may be non-convex. However, we provided conditions under which the optimization problem may still be well-approximated by simple gradient ascent. In future work, we plan on investigating specific real-world applications where it is important to combine expert knowledge with data, such as medical expert systems, and where the expert can give constraints different from what we discussed in this paper.Although, we only defined our priors in the context of Markov logic, most of our results can be generalized for exponential families in a straightforward way.

10 Acknowledgments

This material is based upon work supported by the DARPA Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181, and by ARO grant W911NF-08-1-0242, ONR grant N00014-11-10417, Intel STCPC and NSF grant IIS-1012017. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of DARPA, the Air Force Research Laboratory (AFRL), ARO, ONR, Intel, NSF or the US government. We would like to thank Hung Bui, Tuyen Ngoc Huynh for their helpful discussions during the problem formulation, Natarajan Shankar, Sam Owre for their help with PCE and valuable feedback, and Patrick Lincoln, David Israel for their support and insights.

References

1. Domingos, P., Lowd, D.: Markov Logic: An Interface Layer for Artificial Intelligence. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers (2009)
2. Kok, S., Sumner, M., Richardson, M., Singla, P., Poon, H., Lowd, D., Wang, J., Nath, A., Domingos, P.: The Alchemy system for statistical relational AI. Technical report, Department of Computer Science and Engineering, University of Washington (2010)
3. Geiger, D., Meek, C.: Graphical models and exponential families. In: Proceedings of Fourteenth Conference on Uncertainty in Artificial Intelligence, Madison, WI. Morgan Kaufmann (August 1998) 156–165
4. Fisseler, J.: Toward markov logic with conditional probabilities. In: FLAIRS Conference. (2008) 643–648
5. Thimm, M., Kern-Isberner, G., Fisseler, J.: Relational probabilistic conditional reasoning at maximum entropy. In: ECSQARU. (2011) 447–458

6. Poon, H., Domingos, P.: Joint unsupervised coreference resolution with markov logic. In: EMNLP, ACL (2008) 650–659
7. Pearl, J.: Probabilistic reasoning in intelligent systems - networks of plausible inference. Morgan Kaufmann series in representation and reasoning. Morgan Kaufmann (1989)
8. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press (2009)
9. Chung, F.R.K., Mumford, D.: Chordal completions of planar graphs. J. Comb. Theory, Ser. B **62**(1) (1994) 96–106
10. Raiffa, H., Schlaifer, R.: Applied statistical decision theory [by] Howard Raiffa and Robert Schlaifer. Division of Research, Graduate School of Business Adminitration, Harvard University, Boston, (1961)
11. Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). 1 edn. Springer (2007)
12. Jain, D., Barthels, A., Beetz, M.: Adaptive Markov Logic Networks: Learning Statistical Relational Models with Dynamic Parameters. In: 19th European Conference on Artificial Intelligence (ECAI). (2010) 937–942
13. Ghosh, S., Shankar, N., Owre, S.: Machine reading using markov logic networks for collective probabilistic inference. In: In Proceedings of ECML-CoLISD, 2011. (2011)
14. Ghosh, S., Shankar, N., Owre, S., David, S., Swan, G., Lincoln, P.: Markov logic networks in health informatics. In: In Proceedings of ICML-MLGC, 2011. (2011)
15. Denker, G., Briesemeister, L., Elenius, D., Ghosh, S., Mason, I., Tiwari, A., Bhatt, D., Hailu, H., Madl, G., Nikbin, S., Varadarajan, S., Bauer, G., Steiner, W., Koutsoukos, X., Levendovsky, T.: Probabilistic, compositional, multi-dimension model-based verification (promise)
16. Poon, H., Domingos, P.: Sound and efficient inference with probabilistic and deterministic dependencies. In: AAAI. (2006)
17. Niculescu, R.S., Mitchell, T.M., Rao, R.B.: Bayesian network learning with parameter constraints. Journal of Machine Learning Research **7** (2006) 1357–1383
18. Campos, C.P., Tong, Y., Ji, Q.: Constrained maximum likelihood learning of bayesian networks for facial action recognition. In: Proceedings of the 10th European Conference on Computer Vision: Part III. ECCV '08, Berlin, Heidelberg, Springer-Verlag (2008) 168–181
19. Geman, S., Geman, D.: Readings in computer vision: issues, problems, principles, and paradigms. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1987) 564–584
20. Li, S.Z.: A markov random field model for object matching under contextual constraints. In: In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. (1994) 866–869
21. Druck, G., Mann, G., McCallum, A.: Learning from labeled features using generalized expectation criteria. In: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval. SIGIR '08, New York, NY, USA, ACM (2008) 595–602
22. Liang, P., Jordan, M.I., Klein, D.: Learning from measurements in exponential families. In: Proceedings of the 26th Annual International Conference on Machine Learning. ICML '09, New York, NY, USA, ACM (2009) 641–648
23. Diecidue, E., Wakker, P., Zeelenberg, M.: Eliciting decision weights by adapting de finetti's betting-odds method to prospect theory. Open Access publications from Tilburg University urn:nbn:nl:ui:12-225938, Tilburg University (2007)