

Knowledge Engineering with Markov Logic Networks: A Review

Dominik Jain

Intelligent Autonomous Systems Group
Technische Universität München
jain@cs.tum.edu

Abstract. Within the realm of statistical relational knowledge representation formalisms, Markov logic is perhaps one of the most flexible and general languages, for it generalises both first-order logic (for finite domains) and probabilistic graphical models. Knowledge engineering with Markov logic is, however, not a straightforward task. In particular, modelling approaches that are too firmly rooted in the principles of logic often tend to produce unexpected results in practice. In this paper, I collect a number of issues that are relevant to knowledge engineering practice: I describe the fundamental semantics of Markov logic networks and explain how simple probabilistic properties can be represented. Furthermore, I discuss fallacious modelling assumptions and summarise conditions under which generalisation across domains may fail. As a collection of fundamental insights, the paper is primarily directed at knowledge engineers who are new to Markov logic.

1 Introduction

In artificial intelligence (AI), the unification of statistical and relational knowledge within a single representation formalisms is an important line of research, for it addresses two of the most pressing needs of AI systems: the ability to deal with the uncertainty inherent in real-world domains and the complex interactions between entities that are relevant to an agent, which can adequately be described using relations. In recent years, we have seen tremendous advances in the field that has emerged as statistical relational learning (SRL) and reasoning [2]. One of the most fundamental principles of knowledge representation formalisms developed in SRL is generalisation across domains (shallow transfer), i.e. models are to represent not a concrete probability distribution but rather a set of general principles that can be applied to an arbitrary set of entities and the relations between them in order to obtain a concrete probabilistic model.

A representation formalism that has garnered much attention – owing to its generality and conceptual simplicity – is Markov logic [7], which generalises both first-order logic and probabilistic graphical models (Markov networks) by attaching weights to formulas in first-order logic. Collectively, the weighted formulas define a template for the construction of a graphical model that specifies a distribution over possible worlds. In essence, the probability of a possible world

is proportional to the exponentiated sum of weights of formulas that are satisfied in that world.

As an example, consider the following Markov logic network (MLN),

$$\begin{aligned}
w_1 = 0.000 & \quad \forall p. \text{female}(p) \\
w_2 = -1.735 & \quad \forall p. \text{vegetarian}(p) \\
w_3 = -1.099 & \quad \forall d. \text{vegetarianDish}(d) \\
w_4 = 1.400 & \quad \forall p. \text{female}(p) \Rightarrow \text{vegetarian}(p) \\
w_5 = 1.300 & \quad \forall p_1, p_2. \text{friends}(p_1, p_2) \wedge \text{vegetarian}(p_1) \Rightarrow \text{vegetarian}(p_2) \\
w_6 = 2.300 & \quad \forall p, d. \text{orders}(p, d) \wedge \neg \text{vegetarian}(p) \Rightarrow \neg \text{vegetarianDish}(d) \\
w_7 = (\text{hard}) & \quad \forall p, d. \text{orders}(p, d) \wedge \text{vegetarian}(p) \Rightarrow \text{vegetarianDish}(d)
\end{aligned}$$

which represents a scenario where people go to a restaurant to order a dish. In addition to the weighted formulas themselves, MLNs typically contain declarations that define the types of entities to which predicates can be applied. In our scenario, the predicates are applicable to *person* and *dish* entities. Furthermore, predicates/relations can be declared as being functional, as this is a particularly common requirement in relational domains. In our example, *orders* is declared as functional, such that every person is required to order exactly one dish.

The weights in the MLN determine the degree to which the formulas they are attached to represent a constraint that is believed to hold. By attaching a zero weight to the first formula, we essentially make no statement about whether being female or not being female is more probable. Most people, however, are not vegetarians, and most meals are not vegetarian; hence the negative weights w_2, w_3 . The positive weight w_4 is intended to express that females are more likely to be vegetarians. The fifth formula considers a social network structure defined by the *friends* relation and states that friends of vegetarians are more likely to themselves be vegetarians. The last two formulas express consumption preferences: Non-vegetarians are inclined to order non-vegetarian dishes; vegetarians always order vegetarian dishes. The latter constraint is hard, meaning that any world that does not satisfy the formula is to have zero probability.

The above MLN seems to be an intuitive representation of this very simple scenario. However, it displays some perhaps unexpected behaviour. First, we observe that, for any given domain of people and dishes, the probability of being female is not 0.5, yet *female* appears only in the first formula, which expresses no inclination in either direction, and in the antecedent of an implication (formula 3). Further, the probability is not only far from 0.5, it also varies depending on the number of people and dishes in the domain. For instance, if there are two dishes that can be ordered, then the probability of being female for cases where the number of persons is one, two and three, we obtain approximately¹ 0.223, 0.211 and 0.204 respectively. For *vegetarian* and *vegetarianDish*, we observe particularly wide variations: The probabilities for a dish being vegetarian are 0.182, 0.120 and 0.07 and the probabilities for a person being a vegetarian are 0.084, 0.044 and 0.020 respectively. While at least some of these observations are certainly obvious if one ponders the underlying mathematical definitions, they

¹ All the inference results reported in this paper were computed with exact methods. Where an approximation is mentioned, it refers to the result being rounded only.

suggest that a careful consideration of MLN semantics is the very foundation of any knowledge engineering endeavour.

The mathematical description of the semantics of Markov logic networks is deceptively simple, yet the task of knowledge engineering in Markov logic networks is, as indicated by the above example, not as straightforward as it may at first appear. The question of how one could manually define a Markov logic network remains largely unaddressed in the literature. Indeed, learning is often the only way to sensibly derive suitable parameters for an MLN. However, I believe that a thorough understanding of how one *could*, in principle, represent particular probabilistic properties in an MLN is imperative to selecting the formulas that are in fact required for a model to at all be able to reflect the desired properties and obtain a model that is, ideally, an adequate reflection of the real world. At present, there is no collection of guidelines that adequately addresses the problem of defining weighted formulas for cases where a manual specification might still be feasible. With this paper, I make the following contributions:

- I provide an in-depth discussion of how simple probabilistic properties can be represented in an MLN.
- I point out fallacious assumptions that result from a naive interpretation of the nature of the transition from logical to probabilistic knowledge. In particular, I discuss the fallacious use of probabilistic implications and describe how the intended pieces of knowledge can be represented by other means.
- I summarise practical issues pertaining to the way in which models in Markov logic – be they learnt from data or manually defined – generalise across domains.

2 Markov Logic Networks

Formally, an MLN L is given by a set of pairs $\langle F_i, w_i \rangle$ [7], where F_i is a formula in first-order logic and w_i is a real-valued weight. For each finite domain of discourse D (set of constants), an MLN L defines a *ground Markov network* $M_{L,D} = \langle X, G \rangle$ as follows:

1. X is a set of boolean variables. For each possible grounding of each predicate appearing in L , we add to X a boolean variable (ground atom). We denote by $\mathcal{X} := \mathbb{B}^{|X|}$ the set of possible worlds, i.e. the set of possible assignments of truth values to the variables in X (see Fig. 1).
2. G is a set of weighted ground formulas, i.e. a set of pairs $\langle \hat{F}_j, \hat{w}_j \rangle$, where \hat{F}_j is a ground formula and \hat{w}_j is a real-valued weight. For each possible grounding \hat{F}_j of each formula F_i in L , we add to G the pair $\langle \hat{F}_j, \hat{w}_j = w_i \rangle$. With each such pair, we associate a *feature* $\hat{f}_j : \mathcal{X} \rightarrow \{0, 1\}$, whose value for $x \in \mathcal{X}$ is 1 if \hat{F}_j is satisfied in x and 0 otherwise, and whose weight is \hat{w}_j .

In practice, we typically use a typed logic and augment the above definition with declarations that define the types of entities a predicate applies to. Furthermore,

a particularly common real-world restriction is to require relations to be functional and MLN implementations support the corresponding declarations as an extension.²

The ground Markov network $M_{L,D}$ specifies a probability distribution over the set of possible worlds \mathcal{X} as follows,

$$P(X = x) = \frac{1}{Z} \exp \left(\sum_{i=1}^{|L|} w_i n_i(x) \right) = \frac{1}{Z} \exp \left(\sum_{j=1}^{|G|} \hat{w}_j \hat{f}_j(x) \right) \quad (1)$$

where $Z = \sum_{x' \in \mathcal{X}} \exp(\sum_i w_i n_i(x')) = \sum_{x' \in \mathcal{X}} \exp(\sum_j \hat{w}_j \hat{f}_j(x'))$ is a normalization constant and $n_i(x)$ denotes the number of true groundings of F_i in x .

The probability of a possible world $x \in \mathcal{X}$ is thus proportional to the exponentiated sum of weights of formulas that are satisfied in x , i.e.

$$P(x) \propto \exp \left(\sum_j \hat{w}_j \hat{f}_j(x) \right) =: \omega(x). \quad (2)$$

With $s(F) := \sum_{x \in \mathcal{X}, x \models F} \omega(x)$, we can calculate the probability of any ground formula F_1 given any other ground formula F_2 as

$$P(F_1 | F_2) = \frac{P(F_1, F_2)}{P(F_2)} = \frac{s(F_1 \wedge F_2)}{s(F_2)}. \quad (3)$$

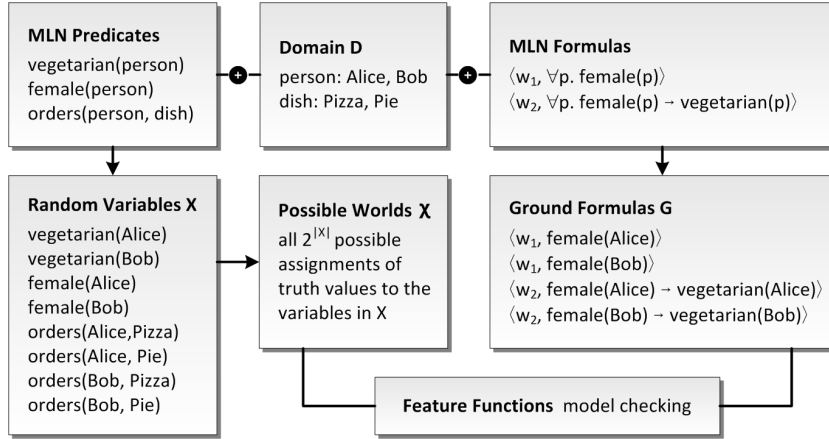


Fig. 1. Illustration of the grounding process

² For instance, the declaration $orders(person, dish!)$ states that the predicate $orders$ is applicable to $person$ and $dish!$ entities and that for each person, there must be exactly one dish for which the predicate holds. Any possible world violating this functional property of the relation has zero probability.

3 Fundamental Semantics

We can differentiate two ways in which MLN semantics can be viewed:

- (V1) the top-down view, in which we seek to realise a probabilistic logic by adding weights to formulas; Markov networks serve only as a formalism that provides the particular probabilistic semantics;
- (V2) the bottom-up view, in which MLNs are an extension of the Markov network formalism and weighted formulas are a particular way of compactly representing generalised features of Markov networks; Any ground formula defines a clique in the ground Markov network and the formula's set of models (satisfying assignments) determines the subset of the clique's configurations that is affected.

From the perspective of artificial intelligence, the top-down view is perhaps more attractive, because the generalisation of logic to probabilistic settings was a long-standing goal of the community. Probability theory is widely accepted as the preferred way of dealing with the inherent uncertainty that permeates real-world domains, and logics are well-understood formalisms for the representation of complex, structured knowledge. To seek a sound unification is, indeed, the merest step of logic.

However, it is the bottom-up view that is imperative to a deep understanding of MLNs. As will be shown in this report, a thorough understanding of Markov networks and, in particular, the way in which weighted formulas translate to features in Markov networks is the very foundation for knowledge engineering in MLNs. Unfortunately, a modelling approach that is too firmly rooted in the principles of logic will often translate sub-optimally to probabilistic settings.

3.1 Quantifier Semantics

A logical knowledge base (KB) is the conjunction of all the formulas that are known to hold. In an MLN, we can choose to partially soften the KB and attach weights to arbitrary parts of the full conjunction, weights indicating the degree to which the individual formulas are required to hold. The granularity at which we apply weighting can be selected as necessary.

The Universal Quantifier for Parameter Sharing. In this context, the semantics of the universal quantifier are particularly relevant. In the logical case, a formula such as $\forall e. \text{apple}(e) \Rightarrow \text{fruit}(e)$ translates, for a finite domain of discourse D , to the conjunction $\bigwedge_{E \in D} \text{apple}(E) \Rightarrow \text{fruit}(E)$. In MLNs, however, the weighted formula $\langle w, \forall e. \text{apple}(e) \Rightarrow \text{fruit}(e) \rangle$ does not result in the weight w being attached to the conjunction but rather in the weight being attached to each conjunct separately, i.e. its semantics can be understood as $\forall E \in D. (\langle w, \text{apple}(E) \Rightarrow \text{fruit}(E) \rangle \in G)$. Thus, the universal quantifier actually serves to realise parameter sharing. Parameter sharing is a fundamental principle in relational models, which serves to achieve the generalisation across

arbitrary domains, for it enables us to apply the same probabilistic patterns to any given set of entities (shallow transfer).

In most implementations of MLNs, a universal quantifier at the outermost level can be omitted – and, given its semantics, its omission may even serve to improve clarity. I consequently chose to omit such universal quantifiers in the remainder of this document.

Existential Quantification. Notably, the existential quantifier does not behave analogously; it expands to a single disjunction with weight w (as does a universal quantifier at an inner level). Hence disjunctions are always evaluated strictly logically. Since KBs are “naturally” dissected into conjunctions, I do not consider this to be a practically relevant restriction, however. Nevertheless, the formalism of recursive random fields [6] was proposed to also allow disjunctions to be evaluated in a “soft” manner.

3.2 Formula Weights

Weights as logarithmized factors. To determine the precise effect of a weight in an MLN, we need only to consider the impact that weights have on the joint distribution over \mathcal{X} (for an arbitrary but fixed grounding). Every non-zero weight w_i of every formula F_i affects the sum $\sum_i w_i n_i(x)$ for some subset of \mathcal{X} and thus has bearing on the distribution. Because $P(X = x)$ is proportional to the *exponentiated* sum of weights that are true in world x , weights are best viewed as logarithmized factors. In the absence of formulas, every possible world’s value $\omega(x)$ is $\exp(0) = 1$ and therefore all worlds are equally probable. If we now add the pair $\langle F_i, w_i = \log(\lambda_i) \rangle$ to L and some possible world x satisfies a ground instance \hat{F}_j of F_i , then $\omega(x)$ is simply multiplied by $\exp(w_i) =: \lambda_i = \hat{\lambda}_j$, either increasing the probability of x (for $\lambda_i > 1 \Leftrightarrow w_i > 0$) or decreasing it (for $\lambda_i \in]0; 1[\Leftrightarrow w_i < 0$). Correspondingly,

$$P(X = x) \propto \prod_j \exp(\hat{f}_j(x) \hat{w}_j) = \prod_{j, x \models \hat{F}_j} \hat{\lambda}_j. \quad (4)$$

When interpreting the semantics of a particular weight w_i of a particular formula F_i , it is imperative to be aware of the factor it implies and the subset of \mathcal{X} it will be applied to (for any imaginable instantiations of L). The impact of a particular formula is perhaps most clearly evident if the formula is translated to disjunctive normal form (DNF), because the conjunctions therein immediately reflect partial assignments in possible worlds. For instance, all of the following are semantically equivalent,

$$\begin{aligned} w & \text{ bird}(e) \Rightarrow \text{flies}(e) \\ w & \neg \text{bird}(e) \vee \text{flies}(e) \\ w & (\text{bird}(e) \wedge \text{flies}(e)) \vee (\neg \text{bird}(e) \wedge \text{flies}(e)) \vee (\neg \text{bird}(e) \wedge \neg \text{flies}(e)) \\ -w & \text{bird}(e) \wedge \neg \text{flies}(e) \end{aligned}$$

yet the latter two forms perhaps more clearly indicate the variable assignments that are affected. In the fourth case, the formula was negated along with its weight to obtain a more easily interpretable conjunction.³

While the local view of the effect of a formula weight on the values $\omega(x)$ associated with possible worlds sufficiently describes the semantics of MLNs, it is, for the most part, not very helpful from a knowledge engineering perspective. In particular, it does not immediately answer the question of how we could define weights that will result in the model reflecting a particular probabilistic property.

Weights as log odds between world probabilities. To determine the effect of a particular weight in terms of probability, let us consider a single (ground) formula \hat{F}_j with an associated weight $w_i = \hat{w}_j$ and let us assume that the MLN contains no further formulas. As mentioned in [7], we can, in such a case, interpret \hat{w}_j as the log odds between the probability of a world where \hat{F}_j is true and the probability of a world where \hat{F}_j is false. Thus, if $x_1 \models \hat{F}_j$ and $x_2 \not\models \hat{F}_j$ ($x_1, x_2 \in \mathcal{X}$), then $P(x_1) : P(x_2) = \hat{\lambda}_j : 1$, since \hat{F}_j being true will result in a factor of $\hat{\lambda}_j$ being applied to worlds satisfying \hat{F}_j while worlds not satisfying \hat{F}_j will keep the default factor of 1. Setting $\hat{w}_j = \log(P(x_1)/P(x_2))$ will thus establish the proper ratio between worlds satisfying \hat{F}_j and worlds not satisfying \hat{F}_j .

Weights that determine formula probabilities. Regarding the probabilistic properties that are to be represented in our model, the local view of ratios between probabilities of individual worlds is clearly inadequate. We need to consider the global impact on all possible worlds in order to assess the properties of the distribution $P(x)$ that are induced by a particular choice of weights, which, in particular, necessitates taking model counts into consideration.

The perhaps most basic modelling task is to define a single formula F_i with an associated weight w_i such that $P(\hat{F}_j) = p$ for some probability p and a ground instance \hat{F}_j of F_i – assuming, for the time being, that F_i is the only formula in the network. If we set $w_i = \log(p_1/p_2)$, then, for $x_1 \models \hat{F}_j$ and $x_2 \not\models \hat{F}_j$, we have that $\omega(x_1) : \omega(x_2) = p_1/p_2 : 1 = p_1 : p_2$ (if x_1 and x_2 are equivalent as far as the truth values of other ground instances of F_i are concerned). To compute the probability $P(\hat{F}_j)$, we need to consider $s(\hat{F}_j)$ and $s(\neg\hat{F}_j)$. With $\#_{\hat{F}_j}$ as the

³ An MLN retains its semantics if any of its formulas are negated along with their weights, i.e. the probability distributions implied by an MLN L and an MLN L' derived from L by changing a pair $\langle F_k, w_k \rangle \in L$ to $\langle \neg F_k, -w_k \rangle \in L'$ are the same for all ground models. If L and L' are instantiated for a set of constants C , for which F_k has N_k groundings, we have

$$\begin{aligned} P_{M_{L',C}}(x) &= \frac{\exp(\sum_{i,i \neq k} w_i \cdot n_i(x) - w_k \cdot (N_k - n_k(x)))}{\sum_{x' \in \mathcal{X}} \exp(\sum_{i,i \neq k} w_i \cdot n_i(x') - w_k \cdot (N_k - n_k(x')))} \\ &= \frac{\exp(\sum_i w_i \cdot n_i(x) - w_k \cdot N_k)}{\sum_{x' \in \mathcal{X}} \exp(\sum_i w_i \cdot n_i(x') - w_k \cdot N_k)} = \frac{\exp(\sum_i w_i \cdot n_i(x))}{\sum_{x' \in \mathcal{X}} \exp(\sum_i w_i \cdot n_i(x'))} \cdot \frac{\exp(-w_k \cdot N_k)}{\exp(-w_k \cdot N_k)} = P_{M_{L,C}}(x) \end{aligned}$$

number of models of the ground formula \hat{F}_j , it follows that if $\#\hat{F}_j = \#\neg\hat{F}_j$, then $s(\hat{F}_j) : s(\neg\hat{F}_j) = p_1 : p_2$ and therefore $P(\hat{F}_j) = p_1/(p_1 + p_2)$. A particularly relevant case where $\#\hat{F}_j = \#\neg\hat{F}_j$ holds is the case where F_i is an atomic formula such as *bird*(*e*). Thus, the probability $P(\text{bird}(e))$ can be set to p for all entities e using

$$\log(p/(1-p)) \quad \text{bird}(e).$$

If, however, the weight w_i applies asymmetrically to the set of possible worlds, i.e. if $\#\hat{F}_j \neq \#\neg\hat{F}_j$, then this has to be taken into consideration. If, for example, the formula \hat{F}_j was satisfied in the majority of possible worlds, then using $\log(p/(1-p))$ as a weight would result in $P(\hat{F}_j)$ being larger than p . To set $P(\hat{F}_j) = p$, we need to find a factor $\hat{\lambda}_j$ such that

$$\begin{aligned} P(\hat{F}_j) &= \frac{s(\hat{F}_j)}{s(\hat{F}_j) + s(\neg\hat{F}_j)} = \frac{\#\hat{F}_j \hat{\lambda}_j}{\#\hat{F}_j \hat{\lambda}_j + \#\neg\hat{F}_j 1} = p \\ \Rightarrow \hat{\lambda}_j &= \#\neg\hat{F}_j p / (\#\hat{F}_j (1-p)) \end{aligned} \quad (5)$$

For instance, if F_i is an implication such as *bird*(*e*) \Rightarrow *flies*(*e*), then we can achieve $P(\text{bird}(e) \Rightarrow \text{flies}(e)) = p$ for all e using

$$\log(p/(3-3p)) \quad \text{bird}(e) \Rightarrow \text{flies}(e)$$

since $\#\hat{F}_j : \#\neg\hat{F}_j = 3 : 1$ in this case.

In all the above considerations, we have made the assumption that the sets of ground atoms appearing in any two ground instances of F_i is disjoint, for if they were to overlap, then there would be interactions between the formulas that might already be too difficult to manually foresee.

Weights of mutually exclusive formulas. There is one simple case where we can easily cope with ground atoms appearing in more than one ground formula – the case where the formulas are mutually exclusive, such that in any possible world, at most one of the formulas is true. For instance, if we have both the formula *bird*(*e*) and the formula $\neg\text{bird}(e)$, then we can achieve $P(\text{bird}(e)) = p$ using

$$\begin{array}{ll} \log(p) & \text{bird}(e) \\ \log(1-p) & \neg\text{bird}(e). \end{array}$$

Because each of the cases is considered in an explicit factor, we need not use odds to establish the desired ratio; the above weighting will immediately ensure that $s(\hat{F}_j) : s(\neg\hat{F}_j) = p : (1-p)$ and therefore $P(\hat{F}_j) = p$ for all instances \hat{F}_j of *bird*(*e*).

This procedure straightforwardly translates to larger sets of mutually exclusive formulas. If, for instance, we have a predicate declared as *isa*(*entity*, *type!*), i.e. an entity belongs to precisely one of several types, then if the set of types is e.g. $\{\text{Mammal}, \text{Bird}, \text{Reptile}\}$ and the corresponding probabilities are p_1, p_2, p_3 , we could set:

$\log(p_1)$ *isa(e, Mammal)*
 $\log(p_2)$ *isa(e, Bird)*
 $\log(p_3)$ *isa(e, Reptile)*

It is desirable for the set of formulas whose weights we set in this way to not only be mutually exclusive but also exhaustive – i.e. exactly one of the formulas should be true for every given world and every binding of the variables – because cases not mentioned in our set of formulas obtain an implicit weight of 0 by default, which in turn implies a higher probability for cases not mentioned, since $0 \geq \log(p_i)$.

It is also important to note that for a mutually exclusive and exhaustive set of formulas, weights are meaningful *only* relative to each other. In particular, for a probability value $p_i \in [0; 1[$ the corresponding weight $\log(p_i)$ is smaller than 0, but this does not at all imply that the corresponding formula being true will lower the probability of worlds satisfying it. In fact, we can add arbitrary offsets $\delta \in \mathbb{R}$ to all of the formula weights without affecting the distribution $P(X = x)$, because if the formulas are mutually exclusive and exhaustive, then the offset will apply to each possible world in exactly the same way: If there are k ground instances of the mutually exclusive and exhaustive set of formulas, then δ is summed k times for each possible world $x \in \mathcal{X}$ and we thus obtain:

$$P(x) = \frac{\exp\left(\sum_j \hat{w}_j \hat{f}_j(x) + k\delta\right)}{\sum_{x' \in \mathcal{X}} \exp\left(\sum_j \hat{w}_j \hat{f}_j(x') + k\delta\right)} = \frac{\exp\left(\sum_j \hat{w}_j \hat{f}_j(x)\right)}{\sum_{x' \in \mathcal{X}} \exp\left(\sum_j \hat{w}_j \hat{f}_j(x')\right)} \cdot \frac{\exp(k\delta)}{\exp(k\delta)} \quad (6)$$

4 The Probabilistic Implication Fallacy

A causal model is often an intuitive representation of a generative stochastic process. Causality dictates a temporal ordering in which values are assigned to variables: If A is the cause of B, then we first determine whether or not A is the case, and depending on the result, we determine whether B is the also case. In the strictly logical case, we can use the implication $A \Rightarrow B$ to represent logical causal influence. In the probabilistic case, the influence we need to model is “If A holds, then B holds with some probability”, which corresponds to the conditional probability $P(B | A)$. The *probabilistic implication fallacy* lies in the assumption that a softened version of the implication $A \Rightarrow B$ is an adequate representation for probabilistic causal influence.

Fallacy 1: Probabilistic implications are equivalent to conditional probabilities. Most likely, one of the reasons for people to think of implications and conditional probabilities as being inextricably linked is that in the strictly logical case, there is indeed an obvious connection. Consider the classical example of modelling the probability with which birds are able to fly. If all birds are able to fly, $P(\textit{flies} | \textit{bird}) = 1$. In this case, the weighted formula

(hard) $\textit{bird}(e) \Rightarrow \textit{flies}(e)$,

which, by definition, implies $P(\textit{bird} \Rightarrow \textit{flies}) = 1$, has related semantics. If *bird* is true, we recover *flies* with probability 1.

In general, however, there is no immediate correspondence between the probability of the implication and the conditional probability. If we set the probability $P(\textit{bird}(e) \Rightarrow \textit{flies}(e))$ to $p \in]0; 1]$, which we can achieve by setting the weight of the implication to $\log(p/(3 - 3p))$ (Eq. 5) or, equivalently, by using the more explicit group of mutually exclusive formulas

$$\begin{array}{ll} \log(p/3) & \textit{flies}(e) \wedge \textit{bird}(e) \\ \log(p/3) & \textit{flies}(e) \wedge \neg\textit{bird}(e) \\ \log(p/3) & \neg\textit{flies}(e) \wedge \neg\textit{bird}(e) \\ \log(1 - p) & \neg\textit{flies}(e) \wedge \textit{bird}(e) \end{array}$$

then we obtain $c := P(\textit{flies}(e) \mid \textit{bird}(e)) = (p/3)/(p/3 + 1 - p) = p/(3 - 2p)$. This expression is equal to p only for $p = 1$ and $p = 0$. For example, if $p = 0.7$, then the conditional probability is actually 0.4375.

Of course, it is possible to set the probability p of the implication such that it will result in any given conditional probability for c (specifically, $p/(3 - 2p) = c \Leftrightarrow p = 3c/(1 + 2c)$). However, as is evident from the above reformulation, the probabilistic implication has bearing not only on the conditional probability of *flies* given *bird* but also on the marginal distributions of both atoms. For instance, for $p = 1$ and $p = 0.7$, we obtain $P(\textit{bird}(e)) = 1/3 = 0.\bar{3}$ and $P(\textit{bird}(e)) = 7/15 = 0.4\bar{6}$ respectively. From the equivalence $\textit{bird}(e) \Rightarrow \textit{flies}(e) \equiv \neg\textit{bird}(e) \vee \textit{flies}(e)$, it should be clear that a greater probability of $\textit{bird}(e) \Rightarrow \textit{flies}(e)$ will result in a lower probability of *bird*(*e*).

Therefore, a probabilistic implication is clearly inappropriate for the representation of a conditional probability.⁴ The probability/weight of an implication does not correspond to a conditional probability in the general case, and implications have influence on the distribution beyond the conditional probability that was intended to be affected.

Fallacy 2: A single implication is sufficient for the representation of probabilistic causal influence. Motivated by the top-down view (V1), it might seem natural to replace the logical influence of *bird* on *flies*, which, in a logical model, might have been represented using the implication $\forall e. \textit{bird}(e) \Rightarrow \textit{flies}(e)$, with a weighted version of the same formula in an MLN. In other words, it might seem natural to make the transition from logical to probabilistic knowledge by simply adding weights to the formulas that are not universally true and thereby make room for exceptions. Unfortunately, this rather intriguing concept turns out to be insufficient if formerly logical dependencies are to be replaced by probabilistic dependencies.

In the strictly logical case, we want to conclude *flies* from *bird*, i.e. if *bird*(*e*) holds, we expect to recover *flies*(*e*) with probability 1. When we make the transition to the probabilistic realm, a causal model will need to represent the degree

⁴ I am, of course, not the first to make this observation, as it essentially follows immediately from the definition of the implication. Joseph Y. Halpern has, for instance, remarked on an analogous issue in his analysis of first-order logics of probability [3].

to which we can conclude *flies* from *bird* and thus represent the conditional probability $P(\textit{flies}(e) \mid \textit{bird}(e))$. As we saw above, the probabilistic implication is a poor candidate for the representation of this probability.

To determine how to best represent such conditional probabilities, a simple analysis suffices. By definition, $P(\textit{flies}(e) \mid \textit{bird}(e)) \propto P(\textit{flies}(e) \wedge \textit{bird}(e))$. The conditional probability is thus concerned with defining the degree to which *flies* and *bird* co-occur. If *bird*(*e*) is given as evidence, we need not concern ourselves with how $P(\textit{bird}(e))$ may be modelled, and by attaching a weight to $\textit{flies}(e) \wedge \textit{bird}(e)$, we can establish any ratio between $\textit{flies}(e) \wedge \textit{bird}(e)$ and $\neg \textit{flies}(e) \wedge \textit{bird}(e)$. In particular, the ratio $p : 1 - p$ can be established using

$$\log(p/(1-p)) \quad \textit{flies}(e) \wedge \textit{bird}(e),$$

setting the conditional probability to p . As long as *bird*(*e*) is given, we could, notably, have used the formula $\textit{flies}(e) \Rightarrow \textit{bird}(e)$ to the same effect. Unfortunately, neither formulation leaves the marginal probability of *bird*(*e*) unchanged, because both affect cases where *bird*(*e*) holds and cases where $\neg \textit{bird}(e)$ holds asymmetrically. The addition of a causal influence (via conditional probabilities) should, however, have no impact on the marginal probability of the cause. Therefore, we need to find weighted formulas that will represent the conditional probability we want to represent whilst guaranteeing that $P(\textit{bird}(e))$ and therefore the ratio $s(\textit{bird}(e)) : s(\neg \textit{bird}(e))$ (for an arbitrary but fixed *e*) remains unaffected. If we consider all the configurations of the clique connecting *bird*(*e*) and *flies*(*e*) explicitly, we will have a maximum of flexibility to satisfy these restrictions.

$$\begin{aligned} w_1 &= \log(\lambda_1) \quad \textit{flies}(e) \wedge \textit{bird}(e) \\ w_2 &= \log(\lambda_2) \quad \neg \textit{flies}(e) \wedge \textit{bird}(e) \\ w_3 &= \log(\lambda_3) \quad \textit{flies}(e) \wedge \neg \textit{bird}(e) \\ w_4 &= \log(\lambda_4) \quad \neg \textit{flies}(e) \wedge \neg \textit{bird}(e) \end{aligned}$$

In order to represent the conditional probability $P(\textit{flies}(e) \mid \textit{bird}(e)) = p$ and thus establish the ratio between $s(\textit{flies}(e) \wedge \textit{bird}(e))$ and $s(\neg \textit{flies}(e) \wedge \textit{bird}(e))$ as $p : 1 - p$, we can simply set $\lambda_1 = p$ and $\lambda_2 = 1 - p$. If, prior to the addition of the four above formulas to our MLN, the model contains no information about the distribution of *flies*(*e*) and therefore $S_B := s(\textit{flies}(e) \wedge \textit{bird}(e)) = s(\neg \textit{flies}(e) \wedge \textit{bird}(e))$ and $S_{\neg B} := s(\textit{flies}(e) \wedge \neg \textit{bird}(e)) = s(\neg \textit{flies}(e) \wedge \neg \textit{bird}(e))$, then $s(\textit{bird}(e)) : s(\neg \textit{bird}(e))$ will remain unaffected if $\lambda_3 + \lambda_4 = \lambda_1 + \lambda_2$, since

$$\frac{s(\textit{bird}(e))}{s(\neg \textit{bird}(e))} = \frac{2 \cdot S_B}{2 \cdot S_{\neg B}} \stackrel{!}{=} \frac{S_B \lambda_1 + S_B \lambda_2}{S_{\neg B} \lambda_3 + S_{\neg B} \lambda_4} \Leftrightarrow \frac{\lambda_1 + \lambda_2}{\lambda_3 + \lambda_4} = 1. \quad (7)$$

The use of complementary probability values for λ_3 and λ_4 trivially satisfies the equation, and the four formulas above will thus collectively represent the entire conditional distribution of *flies* given *bird*. Therefore, if unwanted side-effects are to be avoided, the representation of a full conditional distribution is perhaps the most obvious solution.

We can represent a direct factorisation of the full joint, i.e. $P(\textit{bird}(e), \textit{flies}(e)) = P(\textit{flies}(e) \mid \textit{bird}(e))P(\textit{bird}(e))$, by adding the marginal probability $P(\textit{bird}(e)) =$

p_b to the model using either $\langle \text{bird}(e), \log(p_b/(1-p_b)) \rangle$ or the mutually exclusive pair of $\text{bird}(e)$ and $\neg\text{bird}(e)$ with weights $\log(p_b)$ and $\log(1-p_b)$. In the latter case, we obtain a model that is completely analogous to a directed model in the sense that it represents precisely the same factors with $Z = 1$.

It has now been established that, in order to model causal influence, the use of implications is certainly not as helpful as it may, at first, have appeared. Indeed, the semantics of implications do not seem to translate well to the probabilistic setting. Nevertheless, it is worth noting that it would have been possible to use implications to achieve precisely the same effect as the four conjunctions above. Specifically, the following two transformations are semantically equivalent:

$$\begin{array}{l|l} -w_1 & (w_2 + w_3 + w_4 - 2w_1)/3 \quad \text{flies}(e) \Rightarrow \neg\text{bird}(e) \\ -w_2 & (w_1 + w_3 + w_4 - 2w_2)/3 \quad \neg\text{flies}(e) \Rightarrow \neg\text{bird}(e) \\ -w_3 & (w_1 + w_2 + w_4 - 2w_3)/3 \quad \text{flies}(e) \Rightarrow \text{bird}(e) \\ -w_4 & (w_1 + w_2 + w_3 - 2w_4)/3 \quad \neg\text{flies}(e) \Rightarrow \text{bird}(e) \end{array}$$

The second transformation even retains the normalisation constant. However, I cannot think of a good reason to prefer them – to the contrary: Since implications are not mutually exclusive, their impact on possible worlds is considerably more difficult to interpret.

Syntactic Sugar. From the above considerations, we can draw an important conclusion: Simply adding weights to the logical formulas we might have used in the non-probabilistic case is unlikely to be sufficient if probabilistic dependencies are to be captured to their full extent. While Markov logic networks *are* a generalisation of logic that soundly integrates probabilistic semantics, thinking in logical terms is, for the most part, a hindrance as far as the probabilistic knowledge engineering process is concerned. In fact, I suggest that logical formulations involving (bi)implications be used primarily for the purpose of specifying hard constraints. I argue that, with regard to probabilistic aspects, formulas are mere syntactic sugar for the specification of a subset of a clique’s configurations, and whether or not the logical form fosters comprehensibility of the effect of a weighted formula is debatable. In particular, I discourage the use of implications in probabilistic contexts because implications are typically associated with a directedness which, in probabilistic settings, may not reflect the expected semantics. I argue that the probabilistic counterpart of an implication is not a probabilistic implication but rather a conditional probability (distribution). For the representation of non-deterministic features, an implication such as $\text{bird}(e) \Rightarrow \text{flies}(e)$ should be used only if the negation $\text{bird}(e) \wedge \neg\text{flies}(e)$ is deemed equally appropriate for the task (where the negated form is perhaps the more readable alternative). The key question to ask is: Is the set of partial configurations affected by an implication truly the one we seek to affect or is it the co-occurrence of antecedent and consequent (as in a conditional probability statement) we are interested in?

5 Shallow Transfer

As mentioned in the introduction, one of the key ideas of first-order probabilistic languages such as MLNs is to represent models that generalise across domains – i.e. the principle of *shallow transfer*. For any particular domain (set of entities), we, ideally, want the model to generate a specific ground model that is “adequate”. In this section, I discuss cases where the transition from one domain to another can affect the distribution in perhaps unexpected ways. In short, *the number of entities may matter significantly*.

Invariant properties of distributions across domains. In many scenarios, it is reasonable to assume that the probability distributions represented by a statistical relational model will possess certain invariants that will hold regardless of the instantiation. For instance, in the introductory restaurant example, it might, for instance, be reasonable to expect that the probability of being female should be 0.5 for all people about whom we do not have any evidence. Indeed, if we assume that the model is to represent a (temporally ordered) generative stochastic process which first generates people along with their gender and subsequently generates further properties and links to other entities depending on the gender, then the probability of being female must remain constant in the absence of evidence. Similarly, an invariant might exist for the probability of a dish being vegetarian.

Consider this simplified version of the introductory example,

$$\begin{array}{ll} \log(0.25/0.75) & \text{vegetarianDish}(d) \\ \log(0.15/0.85) & \text{vegetarian}(p) \\ \text{(hard)} & \text{orders}(p, d) \wedge \text{vegetarian}(p) \Rightarrow \text{vegetarianDish}(d) \end{array}$$

where the *orders* predicate is declared as functional (*orders(person, dish!)*, see footnote 2). As we already know from Section 4, the third formula will have influence on the marginal probability of a dish being vegetarian. Therefore, it is not surprising to learn the probability will be greater than the probability 0.25 that is indicated by the first formula. Perhaps more surprising is the fact that the influence of the hard formula varies with the number of entities in the domain. For example, the marginal probability of a given dish being vegetarian is approximately 0.28 for a domain where there are two people and 0.39 for a domain where there are four people and one dish.

The variations are due to combinatorial effects [5]. Without the third formula, if there are N_P persons and N_D dishes, then there are $\alpha(N_P, N_D) := 2^{N_P+N_D} \cdot N_D^{N_P}$ possible worlds with non-zero probability (given that every person consumes exactly one dish). For any pair of possible worlds $\langle x, x' \rangle$ where x and x' differ only in the truth value of *vegetarianDish(d)* for some dish d ($x \models \text{vegetarianDish}(d)$), $\omega(x) : \omega(x') = 25 : 75$. The inclusion of the hard constraint causes the probability of $\delta(N_P, N_D) := \sum_{v=1}^{N_P} \binom{N_P}{v} \sum_{m=1}^v \binom{v}{m} \sum_{i=1}^{N_D} \binom{N_D}{i} i^m (N_D - i)^{v-m} N_D^{N_P-v}$ worlds violating the constraint to be set to zero (v is the number of vegetarians, m is the number of meat-eating vegetarians, and i is the number of non-vegetarian dishes), such that for some pairs $\langle x, x' \rangle$, the ratio $\omega(x) : \omega(x')$

becomes 25 : 0, thus causing worlds with vegetarian dishes to become more probable. Since the fraction $\delta(N_P, N_D)/\alpha(N_P, N_D)$ increases as either N_P or N_D increases, $P(\text{vegetarianDish}(d))$ varies with the number of people and dishes accordingly.⁵

There is, unfortunately, no way of adjusting the parameters of the model such that it will invariantly compute a fixed marginal probability for *vegetarianDish*. Any adjustments made to the weight of the first formula will correct the probability only for a given number of entities.

Moreover, even if we replace the third constraint with the full conditional distribution of *orders* given *vegetarian* and *vegetarianDish* (which would, by definition, leave the marginals unchanged as explained in Section 4), there will still be interactions with domain size owing to the constraint that requires exactly one dish to be consumed by each person. Therefore, in conclusion, there are probabilistic invariants that simply cannot be represented. Because the notion of invariants that are to hold in all instantiations is all but far-fetched (any model of a stochastic process that is causally ordered will typically exhibit invariants), it is a key issue of knowledge engineering, and I deem it essential to be aware of the ramifications.

Several concepts have been introduced to address such issues. For instance, MLNs can be augmented with constraints on formula probabilities, which, in an online adaptation step, can be used to compute corresponding formula weights that will satisfy these constraints [5]. In [1], the notion is extended to constraints on conditional probabilities. Because a computationally expensive online adjustment of parameters may be infeasible, adaptive Markov logic networks were proposed in order to allow weights to be directly represented as functions of domain-specific parameters, which can be learnt from multiple training databases encompassing domains of variable size [4].

Parameter learning may not guarantee that shallow transfer will work as expected. From the analyses above, it should be clear that the manual engineering of MLNs is inherently difficult. As soon as we depart from the pattern of modelling marginal and conditional distributions using mutually exclusive and exhaustive formulas, the interactions between formulas will typically be impossible to predict intuitively. Learning parameters from data (or even the structure of the model) is often advisable. However, it is important to realise that learning may not guarantee that shallow transfer will work as expected. Consider the following MLN,

⁵ Note that there is an analogous effect for the predicate *vegetarian*. For reasons of brevity, its analysis is left to the reader.

w_1	$w_1 + \delta$	$\text{vegetarian}(p)$
w_2	w_2	$\text{vegetarianDish}(d)$
w_3	$w_3 - \delta/N_D$	$\text{orders}(p, d) \wedge \text{vegetarian}(p) \wedge \text{vegetarianDish}(d)$
w_4	$w_4 - \delta/N_D$	$\neg\text{orders}(p, d) \wedge \text{vegetarian}(p) \wedge \text{vegetarianDish}(d)$
w_5	$w_5 - \delta/N_D$	$\text{orders}(p, d) \wedge \text{vegetarian}(p) \wedge \neg\text{vegetarianDish}(d)$
w_6	$w_6 - \delta/N_D$	$\neg\text{orders}(p, d) \wedge \text{vegetarian}(p) \wedge \neg\text{vegetarianDish}(d)$
w_7	w_7	$\text{orders}(p, d) \wedge \neg\text{vegetarian}(p) \wedge \text{vegetarianDish}(d)$
w_8	w_8	$\neg\text{orders}(p, d) \wedge \neg\text{vegetarian}(p) \wedge \text{vegetarianDish}(d)$
w_9	w_9	$\text{orders}(p, d) \wedge \neg\text{vegetarian}(p) \wedge \neg\text{vegetarianDish}(d)$
w_{10}	w_{10}	$\neg\text{orders}(p, d) \wedge \neg\text{vegetarian}(p) \wedge \neg\text{vegetarianDish}(d)$

which is capable of representing a factorisation analogous to a directed model: The first two formulas can represent the marginal distributions of *vegetarian* and *vegetarianDish*, and the remaining formulas can represent the four conditional distributions of *orders* given *vegetarian* and *vegetarianDish*. For a case where there are no constraints on the *orders* relation (such as the requirement that exactly one dish must be consumed by everyone), this set of formulas will allow us to represent invariant marginals for both *vegetarian* and *vegetarianDish* and, by assigning a hard negative weight to the fifth formula and a weight of $0 = \log(1.0)$ to the sixth, it also allows us to represent the “vegetarian constraint”.

However, when learning the weights from data, we will not necessarily recover the invariants – regardless of the degree to which the empirical frequencies in the data match the true parameters (and regardless of the number of independent training databases we may use). This is due to the fact that the learning problem is often ill-posed [5]. For the above MLN, there is an infinite number of weight vectors that represent precisely the same distribution – for a given number of entities. In particular, if the number of dishes is N_D , then for any $\delta \in \mathcal{R}$, the two weight vectors listed above are equivalent and represent precisely the same distribution over possible worlds. As soon as the number of dishes changes, however, the distributions will differ significantly [5]. In our example, the marginal probability of being a vegetarian will vary widely.

A maximum likelihood learner will have no preference of one weight vector over the other. By applying a form of regularisation (e.g. using Gaussian 0-mean priors on the weights as proposed in [7]), we ensure that the optimisation process returns a unique solution. However, that solution is not necessarily the one we expect. Indeed, the solution that results can, as far as generalisation across domains is concerned, be regarded as arbitrary. Hence the soundness of shallow transfer cannot be taken for granted. Indeed, introducing additional knowledge in the form of constraints is necessary in order for the learning process to determine weights that will ultimately represent what is intended. From a knowledge engineering perspective, this is an important point, for it suggests that the task of knowledge engineering in MLNs may require considerations that go beyond the selection of the “right” formulas even when the parameters are learnt from data.

6 Conclusion

In this paper, I scrutinised a number of knowledge engineering questions that arise in Markov logic networks (MLNs). While the collection is by no means complete, I have, nevertheless, addressed many issues that are of high practical relevance. I described the fundamental semantics of MLNs and explained the considerations that are necessary for the representation of simple probabilistic properties with MLNs. With the probabilistic implication fallacy, I explicitly treated one of the most common sources of error, explaining the inherent predicament in softening logic as well as its resolution. Finally, I discussed and summarised conditions under which generalisation across domains can be problematic even for cases where models are learnt from data.

References

1. J. Fisseler. Toward Markov Logic with Conditional Probabilities. In *FLAIRS Conference*, pages 643–648, 2008.
2. L. Getoor and B. Taskar. *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2007.
3. J. Y. Halpern. An Analysis of First-Order Logics of Probability. *Artificial Intelligence*, 46:311–350, 1990.
4. D. Jain, A. Barthels, and M. Beetz. Adaptive Markov Logic Networks: Learning Statistical Relational Models with Dynamic Parameters. In *19th European Conference on Artificial Intelligence (ECAI)*, pages 937–942, 2010.
5. D. Jain, B. Kirchlechner, and M. Beetz. Extending Markov Logic to Model Probability Distributions in Relational Domains. In *30th German Conference on Artificial Intelligence (KI-2007)*, pages 129–143, 2007.
6. D. Lowd and P. Domingos. Recursive Random Fields. In *20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 950–955, 2007.
7. M. Richardson and P. Domingos. Markov Logic Networks. *Machine Learning*, 62(1-2):107–136, 2006.