Fine-Grained Explanations using Markov Logic

Khan Mohammad Al Farabi[1] ✉, Somdeb Sarkhel[2], Sanorita Dey[3], and Deepak Venugopal[1]

[1] University of Memphis
{kfarabi,dvngopal}@memphis.edu
[2] Adobe Research
sarkhel@adobe.com
[3] University of Illinois at Urbana-Champaign
sdey4@illinois.edu

Abstract. Explaining the results of Machine learning algorithms is crucial given the rapid growth and potential applicability of these methods in critical domains including healthcare, defense, autonomous driving, etc. In this paper, we address this problem in the context of Markov Logic Networks (MLNs) which are highly expressive statistical relational models that combine first-order logic with probabilistic graphical models. MLNs in general are known to be interpretable models, i.e., MLNs can be understood more easily by humans as compared to models learned by approaches such as deep learning. However, at the same time, it is not straightforward to obtain human-understandable explanations specific to an observed inference result (e.g. marginal probability estimate). This is because, the MLN provides a lifted interpretation, one that generalizes to all possible worlds/instantiations, which are not query/evidence specific. In this paper, we extract *grounded*-explanations, i.e., explanations defined w.r.t specific inference queries and observed evidence. We extract these explanations from importance weights defined over the MLN formulas that encode the contribution of formulas towards the final inference results. We validate our approach in real world problems related to analyzing reviews from Yelp, and show through user-studies that our explanations are richer than state-of-the-art non-relational explainers such as LIME.

1 Introduction

Markov Logic Networks (MLNs) [1] are popular Statistical Relational Models that combine first-order logic with probabilistic graphical models [10]. The power of MLNs comes from the fact that they can represent relational structure as well as uncertainty in a highly compact manner. Specifically, an MLN represents real-world knowledge in the form of weighted first-order logic formulas. Unlike traditional first-order logic based representations, MLNs allow uncertainty in the represented knowledge, where weights attached to the formulas encode this uncertainty. Larger weights indicate more belief in a formula as compared to smaller weights. The MLN defines a template that can be grounded with real-world constants, to obtain a probability distribution over *possible worlds*

- an assignment to all ground variables - of the MLN. Due to its generality, MLNs have found applications in varied practical problems such as coreference resolution [13], information extraction [12, 23], question answering [8], event-detection in videos [22], etc.

One of the key advantages of MLNs is their interpetability. Specifically, since MLN models are first-order logic based models, it is quite easy for a human user to understand and interpret what the learned model represents. In contrast, methods such as deep learning can achieve state-of-the-art results in language processing, computer vision, etc., but their lack of interpretability is problematic in many domains. However, interpretability of learned models is not the same as explainability of results generated by a Machine learning method. Guidotti et al. [5] provide a detailed survey of explanations in ML methods in which they categorize explanations as model explanations and outcome explanations. The former provides explanations for the model (interpretability of the model) while the latter provides explanations for predictions. Of late, there has been a lot of interest in outcome explanations [6]. For instance, in healthcare applications, a doctor would require a system that explains why it is recommending a particular action, rather than just provide results as a "black-box". Some ML methods such as decision trees are both interpretable and explainable, while some are neither (e.g. deep networks). It turns out that MLNs though interpretable are not easily explainable. Recently proposed approaches such as LIME [15] try to explain the results of a classifier whose results are typically hard-to-understand. However, these approaches are specific to non-relational data, and do not provide rich, relational explanations (for e.g. LIME explains non-linear classifiers as linear models). Our focus in this paper is to explain relational inference in MLNs in a human-understandable form.

Our main idea is to generate explanations for queries in terms of a ranking of formulas based on their importance. Specifically, MLN formulas have weights attached to them that intuitively signify their importance, i.e., for a formula f with weight w, a world where f is true is e^w more likely than a world in which it is false [1]. Note that the formula weights do not have a well-defined probabilistic interpretation if they are dependent on each other, i.e., if atoms in one formula also occur in other formulas [1]. More importantly, the weights are *tied*, which means that any instantiation of a formula has the same weight. Thus, a naive explanation for a query that can be obtained by ranking formulas purely on their weights is not likely to be useful since it is generic across all possible worlds. That is, the explanation will remain unchanged even when the query or evidence variables change. For example, consider the task of classifying if an email is spam or not. An MLN could encode a formula such as $\texttt{Word}(e, +w) \Rightarrow \texttt{Spam}(e)$. The + symbol preceding a variable is a short-hand representation to denote that the MLN stores a different weight for every distinct grounding of the w variable (which represents the domain of words). Suppose the query predicate is `Spam`, we would want different explanations for different groundings of the query predicate based on the specific evidence on the `Word` predicate. Further, suppose the evidence is incomplete, meaning that there are some atoms that are

not query atoms and whose truth value is not known. For formulas containing such atoms, it becomes even harder to determine their influence on a query since we need to consider all possible worlds where the unknown atoms are true as well as the cases where the atoms may be false. We propose a systematic approach for explanations where we learn importance weights for formulas based on samples generated from the MLN. Specifically, we perform inference using Gibbs sampling, and learn the importance of formulas for a specific query based on their influence in computing the Gibbs transition probability. Thus, as the sampler samples possible worlds consistent with the observed evidence, the importance weights capture the influence of formulas on the query variable in these worlds.

We evaluate our approach using two MLN applications we designed for performing inference in real-world review data from Yelp. In the first application, we predict if a review is a spam review and provide explanations for this prediction. In the second application, we predict the sentiment of a review that has missing words. For both cases, we develop MLNs that encode common knowledge and use our approach to extract explanations from the MLNs. We set up a comprehensive user-study consisting of around 60 participants and compare our explanations with explanations given by LIME for the same tasks. We clearly demonstrate through these studies that our explanations are richer and more human-understandable than the explanations given by LIME.

2 Background

2.1 Markov Logic Networks

Markov logic networks (MLNs) are template models that define uncertain, relational knowledge as first-order formulas with weights. Larger the weight of a formula, more likely is that formula to be true. ∞ weight formulas are treated as hard constraints which should always be true. Similarly formulas with $-\infty$ weights are always false. Thus, MLNs offer a flexible framework to mix hard and soft rules. Given a set of constants that represent the domains of variables in the MLN, an MLN represents a factored probability distribution over the possible worlds, in the form of a Markov network. A world in an MLN is an assignment of 0/1 to all ground atoms of the MLN (first order predicates in the MLN whose variables have been substituted with a constant). Specifically, the MLN distribution is given by,

$$P(\omega) = \frac{1}{Z} \exp\left(\sum_i w_i N_i(\omega)\right) \tag{1}$$

where w_i is the weight of formula f_i, $N_i(\omega)$ is the number of groundings of formula f_i that evaluate to True given a world ω, and Z is the normalization constant.

As a simple example of an MLN, suppose we want to encode the fact that smokers and asthmatics are not friends. We would design an MLN with a

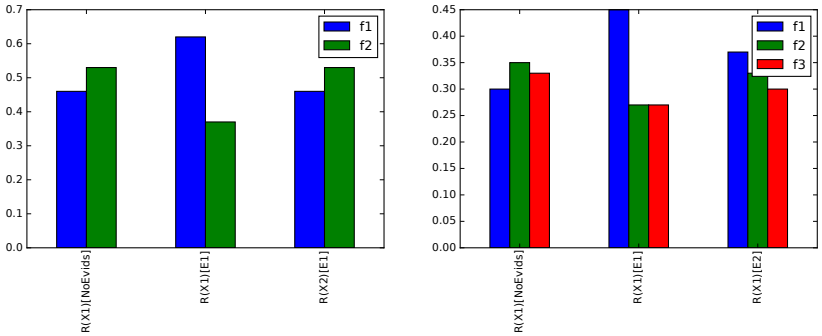
formula such as $\texttt{Smokes}(x) \wedge \texttt{Friends}(x, y) \Rightarrow \neg Asthma(y)$. Given constants corresponding to the variables, x and y, the MLN represents a joint distribution over all ground atoms of `Smokes`, `Friends`, and `Asthma`. The two key tasks in MLNs are *weight learning*, which is the task of learning the weights attached to the formulas from a training relational database, and *inference* (prediction). Learning the weights of an MLN is typically based on Max-likelihood estimation methods. The marginal estimation inference task involves computing the marginal probability distribution of a ground atom in the MLN given an evidence database of observed variables. For example computing the probability that $\texttt{Smokes}(Ana)$ is true given that $\texttt{Smokes}(Bob)$ is true, $\texttt{Friends}(Ana, Bob)$ is true and $\texttt{A}sthma(Bob)$ is false. Since computing this probability exactly is hard, one of the most popular approaches is to use Gibbs sampling [4] to approximate the marginal probability.

2.2 Related Work

Explaining the results of Machine learning models has been recognized as a critical area. Guidotti et al. [5] provide a detailed survey of explanations in ML. Specifically, they categorize them (among others categories) into model explanations and outcome explanations. The former provides explanations for the model while the latter provides explanations for predictions. In this paper, we are primarily concerned with outcome explanations. Recently, there have been a few significant attempts to develop model-agnostic outcome explanations. Notable among these are LIME developed by Ribeiro et. al. [15] which can provide an explanation of *any* classifier, by approximating it locally with an interpretable model. More recently, they developed "Anchors" [16], a model-agnostic explainer with *if-then* rules. Ross et al. [17] developed a regularizer to obtain simpler explanations of a classifier's decision boundary. Koh and Liang [9] addressed the explainability problem by perturbing the importance of training examples and observing their influence on prediction. Similarly, Fong and Veladi [3] also use perturbations to explain predictions. Teso and Kersting [21] recently developed explanations for interactive learners. Though neural networks suffer from lack of interpretability in general, there have been attempts to explain the model through visual analytics, such as Grad-CAM [18] and the more recent work by Zhang and Zhu [24]. However, none of these techniques are applicable to relational data which is the focus of this paper. Specifically, in relational data there is a single example that is interconnected, and is therefore fundamentally different from the type of data addressed in the aforementioned methods. Related to propositional probabilistic graphical models, more recently, Shih et al. [20] compiled Bayesian networks into a more interpretable decision tree model.

3 Query Explanation

Our approach is to extract explanations for a query as a ranked list of MLN formulas, where the ranking encodes the influence of the formula on that query. Before we formally describe our approach, we motivate it with an illustrative



(a) (b)

Fig. 1. Illustrating the influence of a formula w.r.t a query atom for varying evidences. The x-axis shows Query [evidence-set] and the y-axis shows the exponentiated sum of weights for satisfied groundings of the first-order formulas (denoted by $f1$, $f2$, $f3$) which signifies the formula's importance for the query.

example. Consider a simple MLN with 2 formulas, $f_1 = \mathtt{R}(x) \wedge \mathtt{S}_1(x)$ with weight equal to 0.5 and $f_2 = \mathtt{R}(x) \wedge \mathtt{S}_2(x)$ with weight equal to 0.6. Let $\mathtt{R}$ be the query predicate, and let the domain of x, $\Delta_x = \{X_1, X_2, X_3\}$. Let us assume that we want to explain the results of marginal inference, meaning that we compute the marginal probabilities of $\mathtt{R}(X_1) \ldots \mathtt{R}(X_3)$. Given no evidence, in every possible world, f_2 has a larger influence than f_1 in computing the probability of that world. Therefore, the marginal probabilities of the atoms $\mathtt{R}(X_1) \ldots \mathtt{R}(X_3)$, are influenced more by f_2 as compared to f_1. We illustrate this in Fig. 1. Here, we show the exponentiated sum of weights for all satisfied groundings in the first-order formula summed over all possible worlds where the query is satisfied. The values obtained for the formulas f_1 and f_2 are normalized and shown in Fig. 1 (a).

However, now consider a second case, where we add evidence $\mathtt{S}_1(X_1)$ and set all other atoms of $\mathtt{S}_1$ and $\mathtt{S}_2$ to false (we refer to this evidence as $E1$ in Fig. 1). We now analyze the influence of the formulas in a subset of possible worlds that are consistent with the observed evidence. Here, f_1 now has greater significance than f_2 for the query $\mathtt{R}(X_1)$, since the observed evidence makes the formula f_1 grounded with X_1 true and the formula f_2 grounded with X_1 false. However, when we consider a different query $\mathtt{R}(X_2)$, the influence of f_1 and f_2 changes. Specifically, the influence of f_1 and f_2 on $\mathtt{R}(X_2)$ is equivalent to the case where we had no evidence. This is because case f_1 and f_2 grounded with X_2 have the same truth assignment due to the evidence. Thus, the same set of formulas can have different influences on different queries.

Now, suppose, we add a third formula, $f_3 = \mathtt{S}_1(x) \wedge \mathtt{S}_2(x)$ with weight 0.7. Since, f_3 has the highest weight, we may be tempted to say that f_3 has maximum influence on the probabilities. However, if we quantify the influence of the formulas as before, we get the results shown in Fig. 1 (b). Note that adding

the formula changes the influence that the other formulas have on the marginal probability. Further, even though f_3 has a higher weight, its influence on the query $\mathtt{R}(X_1)$ is in fact smaller than that of f_2, even in the case where we have no evidence. Thus, we cannot analyze weights of the formulas independently of each other when the atoms are shared among different formulas, since the weights on one formulas can affect the other formulas.

On adding evidence as specified before, the influence of all three formulas are modified as shown. Further, if we assume a different evidence (specified as $E2$ in Fig. 1) where $\mathtt{S}_1(X_1)$ is true and the other atoms of $\mathtt{S}_1$ and $\mathtt{S}_2$ are unknown (they can be either true or false), then f_3 has a larger influence than the other formulas. Thus, depending upon the evidence as well as the specific query we are looking at, each formula has a different influence on the overall marginal probability. For small examples such as the aforementioned one, we can go over each possible world that is consistent with the evidence and the query, and compute the influence of each formula on the marginal probability of the query. However, this is not practically feasible for large problems. Therefore, we develop a practically feasible solution where we compute the importance based on samples drawn from the distribution over the possible worlds.

To formalize the above example, we first start with some notation. Let $f_1 \ldots f_k$ be the k formulas in the input MLN $\mathcal{M}$. Let $w_1 \ldots w_k$ be weights associated with each of these formulas respectively. Let $\mathbf{Q}$ represent the query predicate, and let $\mathbf{E}$ represent the set of evidence atoms (atoms whose truth assignment is known). Let $q_1 \ldots q_m$ denote the instantiations or ground atoms corresponding to the query predicate. Note that, for the sake of of clarity, we assume that we have a single query predicate, however, it is straightforward to include multiple query predicates.

3.1 Sampling

In standard Gibbs sampling for MLNs, we start with a random assignment to all atoms $\omega^{(0)}$ in the MLN except the evidence atoms whose assignments are fixed as given in $\mathbf{E}$. In each iteration of Gibbs sampling, we choose a non-evidence atom based on a proposal distribution α, and compute an assignment to this atom by sampling the assignment based on its conditional distribution. In our case, we assume that α is a uniform distribution, which means that we sample non-evidence atoms randomly in each iteration. From the generated samples, we estimate the marginal probabilities of $P(q_1) \ldots P(q_m)$ as,

$$P(\bar{q}_i) = \frac{1}{T} \sum_{t=1}^{T} \mathbb{I}(\omega^{(t)} \sim \bar{q}_i) \tag{2}$$

where T is the total number of samples, $\omega^{(t)} \sim \bar{q}_i$ denotes that the assignment to atom q_i in $\omega^{(t)}$ is consistent with $\bar{q}_i$. Without loss of generality, we assume that $\bar{q}_i$ refers to the true (or 1) assignment to q_i. Thus, to compute the marginal probability for q_i, we need to compute the ratio of the number of samples where the q_i was equal to true (or 1) and the total number of samples collected.

Suppose we choose to sample a query atom, q_i in an iteration of Gibbs sampling, the main task is to compute the conditional distribution $P(q_i|\omega^{(t-1)}\backslash q_i)$, where $\omega^{(t-1)} \setminus q_i$ is the set of assignments to all atoms except q_i in the sample at iteration $t-1$. Once we compute the conditional distribution, we sample the assignment for q_i, say $\overline{q}_i$ from the distribution, and the subsequent sample $\omega^{(t)} = \omega^{(t-1)} \cup \overline{q}_i$. The conditional distribution to be computed in an iteration is given by,

$$P(\bar{q}_i|\omega^{(t-1)} \setminus q_i) = \exp \sum_j w_j N_j(\omega^{(t-1)} \setminus q_i \cup \bar{q}_i) \tag{3}$$

where $N_j(\omega^{(t-1)} \setminus q_i \cup \bar{q}_i)$ is the number of satisfied groundings in the j-th formula given the assignment $\omega^{(t-1)} \setminus q_i \cup \bar{q}_i$.

We now define the importance distribution for a query atom q_i, $\mathcal{Q}(q_i)$ as follows. In each step of Gibbs sampling, where q_i is satisfied, we measure the contribution of each formula to the Gibbs transition probability. Specifically, for a formula f_k, its contribution to the transition probability is proportional to $\exp(w_j N_j(\omega^{(t-1)} \setminus q_i \cup \bar{q}_i)$, if q_i is the atom being sampled in iteration t. However, since we consider both cases in the conditional probability, namely, the assignment 1 (or true) to $\bar{q}_i$ as well as the assignment 0 (or false) to $\bar{q}_i$, we would like to encode both these into our importance function. To do this, we compute the log odds of a query atom, and score the influence of a formula on the query based on its contribution in computing its log-odds.

Formally, let $\omega^{(t-1)}$ be the Gibbs sample in iteration $t-1$. Suppose we are sampling the query atom q_i, we compute the log-odds ratio between the Gibbs transition probability for $q_i = 0$ and $q_i = 1$. This is given by the following equation,

$$\begin{aligned} \log \frac{P(q_i = 1|\omega^{(t-1)} \setminus q_i)}{P(q_i = 0|\omega^{(t-1)} \setminus q_i)} = \\ \sum_j w_j N_j(\omega^{(t-1)} \setminus q_i \cup \{q_i = 1\}) \\ - \sum_j w_j N_j(\omega^{(t-1)} \setminus q_i \cup \{q_i = 0\}) \end{aligned} \tag{4}$$

$$\begin{aligned} \log \frac{P(q_i = 1|\omega^{(t-1)} \setminus q_i)}{P(q_i = 0|\omega^{(t-1)} \setminus q_i)} = \\ \sum_j w_j (N_j(\omega^{(t-1)} \setminus q_i \cup \{q_i = 1\}) \\ - N_j(\omega^{(t-1)} \setminus q_i \cup \{q_i = 0\}) \end{aligned} \tag{5}$$

We then update the importance weight of the j-th formula w.r.t query q_i as

$$\mathcal{Q}_j^{(t)}(q_i) \propto w_j N_j(\omega^{(t-1)} \setminus q_i \cup \{q_i = 1\}) - w_j N_j(\omega^{(t-1)} \setminus q_i \cup \{q_i = 0\}) \tag{6}$$

We update all the importance weights for q_i denoted by $\mathcal{Q}^{(t)}(q_i) = \mathcal{Q}_1^{(t)}(q_i)$, $\ldots$ $\mathcal{Q}_k^{(t)}(q_i)$ corresponding to the formulas 1 through k in every iteration where q_i is sampled. The importance weight for $\mathcal{Q}_j^{(t)}(q_i)$ after sampling q_i T times is given by,

$$\mathcal{Q}_j(q_i) = \frac{1}{T}\sum_{t=1}^{T}\mathcal{Q}_j^{(t)}(q_i) \tag{7}$$

Theorem 1. *As $T \to \infty$,*

$$log\frac{P(q_i=1)}{P(q_i=0)} \propto \sum_j \mathcal{Q}_j(q_i) \tag{8}$$

Proof.

$$\begin{aligned} log\frac{P(q_i=1)}{P(q_i=0)} = & \\ & \sum_\omega \log\frac{P(\omega \sim q_i=1)}{P(\omega \sim q_i=0)} \\ & \propto \sum_\omega \sum_j w_j(N_j(\omega \sim q_i=1) \\ & - N_j(\omega \sim q_i=0)) \\ & \propto \sum_\omega \sum_j w_j(N_j(\omega \sim q_i=1) \\ & - \sum_j N_j(\omega \sim q_i=0)) \end{aligned} \tag{9}$$

where $\omega \sim q_i = 1$ are worlds consistent with the known evidence as well as $q_i = 1$, and $\omega \sim q_i = 0$ are worlds consistent with the known evidence $q_i = 0$. Further

$$\mathbb{E}[\mathcal{Q}_j(q_i)] = \sum_\omega w_j(N_j(\omega \sim q_i=1) - w_j N_j(\omega \sim q_i=0)) \tag{10}$$

as $T \to \infty$, $\mathcal{Q}_j^{(t)}(q_i) \to \mathbb{E}[\mathcal{Q}_j(q_i)]$, since we are estimating the expectation from worlds consistent with the MLN distribution. Therefore, as $T \to \infty$, $\sum_j \mathcal{Q}_j^{(t)}(q_i)$ $\sum_j \mathbb{E}[\mathcal{Q}_j(q_i)]$ which is equal to the log-odds ratio $log\frac{P(q_i=1)}{P(q_i=0)}$

Interestingly, it turns out that in some cases, the importance weights can be obtained without sampling multiple worlds. Specifically, we can show that,

Proposition 2. *If the evidence is complete, i.e., every non-query atom is known to be either true or false, and if every ground formula in the MLN contains exactly one query atom, then $\mathbb{E}[\mathcal{Q}_j(q_i)] = w_j N_j(\omega \sim q_i = 1) - w_j N_j(\omega \sim q_i = 0)$, where ω is any world consistent with the known evidence.*

Algorithm 1: Explaining Inference

Input: MLN $\mathcal{M}$, Evidence $\mathbf{E}$, Query atoms $\mathbf{Q}$
Output: Ranking of formulas in $\mathcal{M}$ for each $q_i \in \mathbf{Q}$

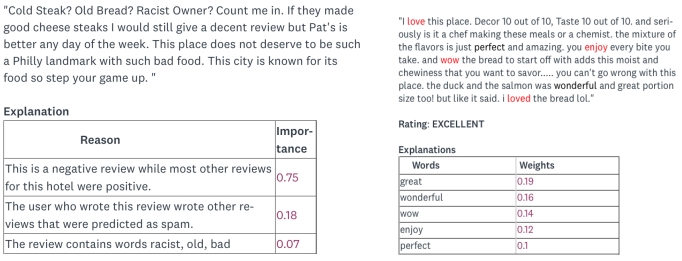
```
Initialize the non-evidence atoms in ω^(0) randomly
for t = 1 to T do
    X = Choose a non-evidence atom in ω^(t) uniformly at random
    Flip X in ω^(t) to compute the conditional distribution P(X|ω^(t) \ X)
    Sample X from P(X|ω^(t) \ X)
    if X ∈ Q then
        for each f_j in M do
            Update the importance weight Q_j^(t)(X)
for each q_i ∈ Q do
    Explain q_i as a ranked list of formulas f_1 ... f_k based on importance
     weights in Q(q_i)
```

The above proposition implies that, in MLNs where the evidence is fully specified over the non-query atoms, and every query atom occurs in an independent subset of ground formulas in the MLN, we can derive the importance weights directly from the specified evidence. However, in cases where the evidence does not cover all the ground atoms, or more than one query atom occurs in a ground formula, we cannot infer its importance without sampling the possible worlds. Note that in general, instead of using Gibbs sampling to generate the possible worlds, we can use Marginal-MAP inference to sum-out the unknown atoms, and then derive the explanations using the evidence. However, marginal-MAP is considerably more expensive [19]. Another strategy is to use the MAP assignment for the unknown atoms. However, this is problematic when we have a significant number of unknown atoms, and if the distribution is multi-modal since, we are essentially considering a single world. A third strategy is to use belief propagation. However, the unknown atoms is again problematic in this case since we need to sum out those atoms to derive the belief propagation messages, and for large number of unknown atoms, this can be extremely expensive. Thus, our sampling strategy allows us to estimate the importance weights in a computationally feasible manner.

Algorithm 1 summarizes our approach. First, we initialize all non-evidence atoms in the MLN randomly. In each iteration, we select a non-evidence atom uniformly at random, and compute the conditional distribution for that atom given the state of all other atoms. Based on this conditional probability, we sample a new assignment for the sampled atom. If the sampled atom is a query atom, for each formula, we compute its importance weight for that query in the current word using Eq. (6). We update the importance weight using Eq. (6). Once the marginal probabilities in the Gibbs sampler converge, we finally compute a explanation for the marginal probability obtained for each query atom by ranking the formulas in descending order of the importance weights specific to that query.

4 Experiments

Our main goal is to evaluate if the explanations output by our approach helps a user understand the "black-box" that is giving this particular explanation. To do this, we designed a comprehensive user study consisting of around 60 participants. We compared our approach with the explanations given by LIME [15], an open-source state-of-the-art explanation system. We perform our evaluation using two real-world tasks on a Yelp dataset [14]. We sampled 1000 reviews from this dataset for our experiments. In the first task, we design an MLN that performs joint inference to predict if a review is filtered as a spam review or not by Yelp. In the second task, we predict if a review has positive or negative sentiment based on the review content. We first describe our user study setup and then present the details of our applications along with the results.



(a) Spam (b) Sentiment

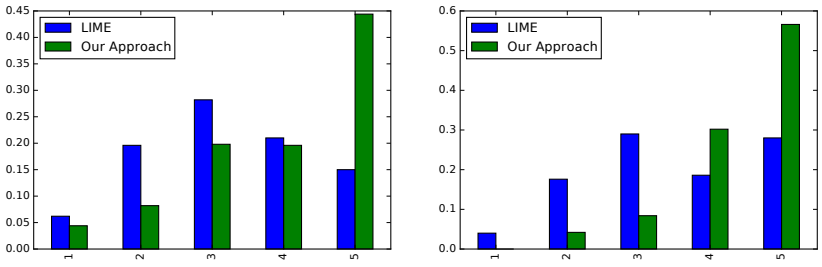
Fig. 2. Explanations generated by our approach. (a) shows the explanations for the spam prediction application and (b) shows the explanation for the sentiment prediction where the red-colored words are considered as hidden/missing words.

4.1 User Study Setup

Our user study group consisted of students who have varying backgrounds in Machine learning. The participants were either enrolled in the Machine learning course at University of Memphis or part of the Machine learning club. The participants included undergraduate students, Master's students as well as Ph.D. students. All of them understand classification algorithms and the basics of Machine learning. A few participants were advanced researchers in related areas including Natural Language Processing, computer vision, etc. We divided the participants into two groups, and sent the survey that had the explanations generated by LIME to one group and the explanations generated using our approach to the other group. To ensure that there was no bias in the results, the users did not know whether they were evaluating LIME or our approach. There were 10 questions in each survey. The first 5 questions asked the participants to rank the explanations on a scale of 1 - 5. The next three questions were used to measure three dimensions of the explanation as follows.

1. Q6: Did the explanations increase your *understanding* of how the classifier is detecting ratings of reviews?
2. Q7: Did the explanations increase your *trust* in the classifier?
3. Q8: Based on the above explanations, will you be able to *apply* this knowledge to predict spam (or sentiment) given a set of new reviews?

Each of the above questions had a response scale of 1 - 5, with 5 being the best score. Finally, we summarized the overall explanation quality by asking participants if they would have liked the classifier to give them more explanations, less explanations or if they felt the explanations provided by the classifier was just right. We also allowed users to enter other comments in free text format.



(a) (b)

Fig. 3. Comparison of LIME and our approach using explanation scores as rated by the users. (a) shows this for the spam prediction application and (b) for the sentiment prediction. In each case, we show the % of users who have given a specific score for an explanation, averaged across all the explanations.

4.2 Application 1: Review Spam Filter

Detecting filtered reviews is a challenging problem. Specifically, unlike say email spam, spam reviews look a lot more authentic since it is designed to influence a user for/against a product/service in an open forum. This task more generally called opinion spam has a large body of prior work starting with work by Jindal and Liu [7]. In this case, we develop an MLN that encodes knowledge for detecting spam, and then perform inference on the MLN while generating the explanations.

Our MLN contains formulas that connect words to predicate that indicates whether they are spam $\texttt{Word}(+w, r) \Rightarrow \texttt{Spam}(r)$. We then add relational information into the MLN. Specifically, given two reviews about the same restaurant, the spammer and non-spammer provide ratings that are opposite of each other. For e.g., a spammer provides a positive or high rating, while a non-spammer provides a negative or low rating. Naturally, this is not always true and is therefore a soft formula in the MLN. Finally, we add knowledge that if given two reviews by the same person, if one of them is predicted spam, the other one is likely to be

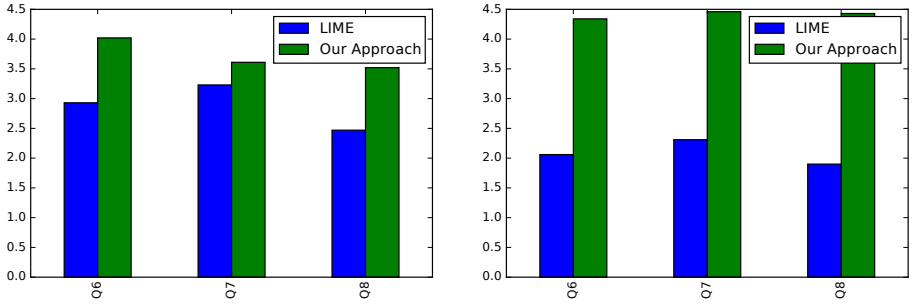
spam as well. In this MLN, note that the evidence variables are the words, we consider the ratings as unknown variables and the query variables are the atoms of predicate `Spam`. Since ratings are unknown, this is a joint inference problem where we infer the rating of a review jointly with inferring if the review is spam or not. We therefore add formulas connecting words with the rating. We learn the MLN by initializing it with weights that we obtain from an SVM [2]. Specifically, we learn an SVM for predicting ratings from the review text, as well as one for predicting if a review is spam/not. Using the coefficients of the MLN, we set initial weights to formulas [2] such as $\texttt{Word}(+w, r) \Rightarrow \texttt{Spam}(r)$, and then use Tuffy [11] to learn the weights of the MLN. The five fold cross validation F1-score using MLNs for this task was around 0.7. We perform inference and generate explanations for the queries. We picked a small sample of query explanations to conduct the user survey.

Once we perform inference and obtain the importance weights of the formulas, we ranked them, and converted the formula into English to generate the human-readable explanation. We presented the user with this explanation as well as the importance weights (normalized) for the 5 most important formulas. An example of the explanation generated is shown in Fig. 2 (a). The users could look at the original review and rate the explanation for that review. For LIME, we provided the input which is the review content and since LIME does not explain relational information, it uses the non-relational features (words/phrases) to come up with its explanation using SVMs as the base classifier.

The comparison of the user response scores for the explanations is shown in Fig. 3. As seen here, on average, across the reviews in the survey, a larger percentage of users gave our explanations higher scores as compared to the explanation generated by LIME. On the other hand, a large percentage of users rated LIME explanations around the halfway mark (score 3). Further, when we analyze the responses over the three explanation dimensions as shown in Fig. 4 (a), we see that our approach was favored by participants in all three dimensions. Particularly, the dimensions of understanding the classifier and being able to use the knowledge in the explanation scored much higher. This shows that including higher-level relational knowledge in the explanations makes the explanations richer and more appealing to humans.

4.3 Application 2: Review Sentiment Prediction

In this application, we predict if a review has has a positive or negative sentiment based on the words in that review. Specifically, we have MLN formulas that connect words in the review to the sentiment. However, we assume incomplete evidence. That is, we remove a small set of words from the review and therefore, their state is unknown. The inference task is to jointly infer the state of the hidden words along with predicting if it is a positive or negative sentiment review. To do this, we add relational knowledge to the MLN. Specifically, we encode MLN formulas that a user is likely to use the same words to describe a positive or negative rating. Thus, we can use words from other reviews written by the same user to predict the sentiment of a review. We learn the MLN using a similar



(a) Spam (b) Sentiment

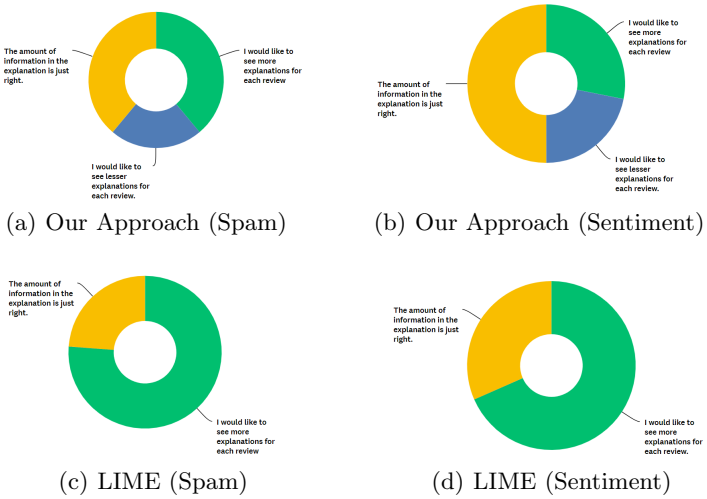
Fig. 4. Comparison for the average scores given by users for 3 key dimensions related to the explanations. Q6 measures understanding of the classifier, Q7 the trust in the classifier and Q8 if they can replicate the classifier based on the explanation. Higher scores are better. (a) shows results for spam prediction and (b) shows results for sentiment prediction

procedure as described in the previous section. Our five-fold cross validation accuracy here was around 0.9.

In this case, we generate explanations in terms of word formulas only. Specifically, for each review, we explain its predicted sentiment as a set of words (and their corresponding importance weights). Note that these words can contain missing words (inferred to be true) as well as words known to be true (due to evidence). Thus, LIME and our approach generates the same form of explanations (words and weights) as shown in Fig 2 (b). However, since we can infer the states of hidden words, our explanation is richer than that generated by LIME. Fig. 3 (b) shows the comparison of the explanation scores for LIME and our approach. Here, we see a very similar trend to the results for the spam prediction application. Specifically, most users thought that our approach yields very good explanations, while LIME explanations was considered average. Further, Fig. 4 (b) illustrates that our approach was significantly better in terms of helping users understand, trust and apply the prediction method. This shows that using relational knowledge can yield a more comprehensive explanation (in the presence of noisy/unknown variables).

4.4 T-Test

We use the T-Test to compare the means of the two user groups (those who evaluated LIME and our approach respectively). The null hypothesis for the t-test is that there is no difference between the means of the two groups. In our case, it will mean that our explanation is no better or worse than the lime explanation. The alternate hypothesis is that the means of the two groups are not the same, in which case it will mean that our explanation is either better or worse than the lime explanation. We performed the t-test on the responses



(a) Our Approach (Spam) (b) Our Approach (Sentiment)

(c) LIME (Spam) (d) LIME (Sentiment)

Fig. 5. Comparison of user responses for .the question that summarizes the effectiveness of explanations. (a) and (b) show this for spam prediction and sentiment prediction using our approach, and (c), (d) show the responses for LIME.

to the summary question regarding the quality of the explanations. We coded these as follows. i) rating for lime explanation (coded as group 1) ii) rating for our explanation (coded as group 2). The response options are, i) would like to see more explanation (coded as 1) ii) would like to see less explanation (coded as 2) and iii) Right amount of explanation (coded as 3). The coding is based on the desirability of the response. We assumed the best case is the right amount of explanation and therefore coded this as the highest. Then, we assumed that requiring less amount of information is worse than right amount of information, and is therefore coded as 2. Finally, we assumed that a user requiring more amount of information is the worst case (coded as 1) because our main motivation is to make the explanation human-interpretable. Thus, according to our coding, the higher mean will be considered better because we coded the right amount of information as the highest. We obtained $p = 0.03$ (< 0.05). Therefore, the difference in explanations provided by our approach is statistically significant. Thus, we can reject the null hypothesis and our explanation is at least better or worse than LIME explanation. The mean and standard deviations of the two approaches (based on the coding) is as follows. *LIME explanation has a mean of 1.63 and a standard deviation of 0.95. The explanations based on our approach has a mean of 2.22 and a standard deviation of 0.87.* Therefore, our explanations are clearly preferred by the users as compared to the explanations given by LIME. The full breakdown of the responses is shown in Fig. 5. As seen here, in each of the two tasks, users considered our explanations to be better than LIME. Interestingly, even in the case where the type of explanations was identical (words explaining the sentiment), LIME produced worse results than our

approach (see Fig. 5 (b) and (d)) because our relational method takes advantage of dependencies across different reviews to generate more complete explanations.

5 Conclusion

Explanations of predictions made by machine learning algorithms is critical in several application domains. In general, MLNs are interpretable models but it is challenging to explain results obtained from inference over MLNs. In this paper, we presented an approach where we explain the results of relational inference in MLNs as a ranked list of formulas that encode their influence on the inference results. Specifically, we compute the importance weights of the MLN formulas based on how much they influence the transition probabilities of a Gibbs sampler that performs marginal inference in the MLN. On two real-world problems, we conducted a comprehensive user study and showed that our explanations are more human-interpretable as compared to explanations derived from LIME, a state-of-the-art approach for explaining classifiers.

In future, we plan to apply our approach to explain complex queries in multi-modal problems, derive alternate forms of explanation that improve interpretability, etc.

References

1. Domingos, P., Lowd, D.: Markov Logic: An Interface Layer For Artificial Intelligence. Morgan & Claypool, (2009)
2. Farabi, M.K.A., Sarkhel, S., Venugopal, D.: Efficient Weight Learning in High-Dimensional Untied MLNs. In: Artificial Intelligence and Statistics (AISTATS). pp. 1637–1645 (2018)
3. Fong, R.C., Vedaldi, A.: Interpretable Explanations of Black Boxes by Meaningful Perturbation. In:International Conference on Computer Vision (ICCV), pp. 3449–3457. (2017)
4. Geman, S., Geman, D.: Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. IEEE Transactions on Pattern Analysis and Machine Intelligence **6**, 721–741 (1984)
5. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A Survey of Methods for Explaining Black Box Models. ACM Comput. Surv. **51**(5), 93:1–93:42 (2018)
6. Gunning, D.: Darpa's Explainable Artificial Intelligence (XAI) Program. In: ACM IUI (2019)
7. Jindal, N., Liu, B.: Opinion Spam and Analysis. In: Proceedings of the International Conference on Web Search and Data Mining. pp. 219–230 (2008)
8. Khot, T., Balasubramanian, N., Gribkoff, E., Sabharwal, A., Clark, P., Etzioni, O.: Exploring Markov Logic Networks for Question Answering. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 685–694 (2015)
9. Koh, P.W., Liang, P.: Understanding Black-Box Predictions Via Influence Functions. In: Proceedings of the 34th International Conference on Machine Learning (ICML). pp. 1885–1894 (2017)

10. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press (2009)
11. Niu, F., Ré, C., Doan, A., Shavlik, J.W.: Tuffy: Scaling Up Statistical Inference in Markov Logic Networks Using an RDBMS. PVLDB **4**(6), 373–384 (2011)
12. Poon, H., Domingos, P.: Joint Inference in Information Extraction. In: Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI). pp. 913–918. AAAI Press (2007)
13. Poon., H., Domingos, P.: Joint Unsupervised Coreference Resolution with Markov Logic. In: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 649–658. (2008)
14. Rayana, S., Akoglu, L.: Yelp Dataset for Anomalous Reviews. Tech. rep., Stony Brook University , `http://odds.cs.stonybrook.edu`.Last accessed 2015
15. Ribeiro, M.T., Singh, S., Guestrin, C.: Why Should I Trust You?": Explaining the Predictions of Any Classifier. In: Knowledge Discovery and Data Mining (KDD). pp. 1135–1144. (2016)
16. Ribeiro, M.T., Singh, S., Guestrin, C.: Anchors: High-Precision Model-Agnostic Explanations. In: AAAI Conference on Artificial Intelligence (AAAI) (2018)
17. Ross, A.S., Hughes, M.C., Doshi-Velez, F.: Right for the Right Reasons: Training Differentiable Models By Constraining Their Explanations. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI). pp. 2662–2670. (2017)
18. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-Cam: Visual Explanations From Deep Networks Via Gradient-Based Localization. In: 2017 IEEE International Conference on Computer Vision (ICCV). pp. 618–626 (2017)
19. Sharma, V., Sheikh, N.A., Mittal, H., Gogate, V., Singla, P.: Lifted Marginal MAP Inference. In: Uncertainty in Artificial Intelligence. pp. 917–926. AUAI Press (2018)
20. Shih, A., Choi, A., Darwiche, A.: A Symbolic Approach to Explaining Bayesian Network Classifiers. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI). pp. 5103–5111 (2018)
21. Teso, S., Kersting, K.: "Why Should I Trust Interactive Learners?" Explaining Interactive Queries of Classifiers To Users. Arxiv (2018)
22. Tran, S.D., Davis, L.S.: Event Modeling And Recognition Using Markov Logic Networks. In: European Conference on Computer Vision (ECCV). pp. 610–623. (2008)
23. Venugopal, D., Chen, C., Gogate, V., Ng, V.: Relieving The Computational Bottleneck: Joint Inference for Event Extraction with High-Dimensional Features. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 831–843. (2014)
24. Zhang, Q., Zhu, S.: Visual Interpretability for Deep Learning: A Survey. Arxiv (2018)