Interpretable Explanations for Probabilistic Inference in Markov Logic

Khan Mohammad Al Farabi
University of Memphis
kfarabi@memphis.edu

Somdeb Sarkhel
Adobe Research
sarkhel@adobe.com

Sanorita Dey
UMBC
sanorita@umbc.edu

Deepak Venugopal
University of Memphis
dvngopal@memphis.edu

2021 IEEE International Conference on Big Data (Big Data) | 978-1-6654-3902-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/BigData52589.2021.9671572

***Abstract*—Markov Logic Networks (MLNs) represent relational knowledge using a combination of first-order logic and probabilistic models. In this paper, we develop an approach to explain the results of probabilistic inference in MLNs. Unlike approaches such as LIME and SHAP that explain black-box classifiers, explaining MLN inference is harder since the data is interconnected. We develop an explanation framework that computes importance weights for MLN formulas based on their influence on the marginal likelihood. However, it turns out that computing these importance weights exactly is a hard problem and even approximate sampling methods are unreliable when the MLN is large resulting in non-interpretable explanations. Therefore, we develop an approach where we reduce the large MLN into simpler coalitions of formulas that approximately preserve relational dependencies and generate explanations based on these coalitions. We then weight explanations from different coalitions and combine them into a single explanation. Our experiments illustrate that our approach generates more interpretable explanations in several text processing problems as compared to other state-of-the-art methods.**

***Index Terms*—Explainable AI, Markov Logic Networks, Statistical Relational Models**

I. Introduction

Explainable Artificial Intelligence (XAI) [9] has made significant progress over the last few years. In particular, several approaches that explain results from black-box classifiers have been developed [19, 14]. However, these explainers typically work for classification problems when the data instances are independent of each other. To explain predictions on relational data, we need to factor in the influence of relationships in the data. In this paper, we develop an approach for explanations in probabilistic inference in Markov Logic Networks (MLNs) [4], a well known statistical relational model. MLNs consists of first-order logic formulas with weights attached to them. However, weights attached to the formulas as such do not have a direct probabilistic interpretation that can be used for explanations. Instead, we need to perform probabilistic inference to quantify the influence of formulas in the distribution. To explain inference in MLNs, we formalize explanations as expected values of formula states. We can then estimate these expectations from samples drawn from the distribution of the MLN using approaches such as Markov Chain Monte Carlo. However, while this approach yields sound explanations, in practice, explaining large MLNs is a hard problem. Specifically, the underlying probabilistic model becomes very large and well-known sampling methods such as Gibbs sampling have poor *mixing* properties in such large MLNs [20]. Consequently, explanations derived from a poorly mixed sampler tend to produce results that have poor interpretability. To improve interpretability of explanations, here, we derive explanations from an approximate MLN. Specifically, we construct *coalitions* of formulas where a coalition tries to maintain the relational structure present in the original MLN. To do this, we leverage symmetries in the MLN, i.e., variables that have similar relational structure to other variables in the MLN. We derive coalitions with a reduced number of variables by sampling from groups of (approximately) symmetric variables in the MLN. We then explain probabilistic inference outcomes in the coalitions.

However, once we have multiple explanations from different coalitions, it turns out that combining these explanations is challenging in our case since it is hard to weight the explanations. Ideally, the weights should encode the distance between the coalition to the true distribution, i.e., give closer approximations of the MLN a larger weight in the overall explanation. However, computing the true distribution in our case is intractable. Therefore, we develop a weighting method that penalizes explanations where the influence of formulas significantly deviates from the MLN parameterization. That is, if the ranking of formulas in the explanation tend to match the rankings based on the MLN weights, then the explanation is more consistent with the MLN parameterization and therefore is given a larger weight. We integrate the explanations by combining the weighted ranking orders in each explanation. Specifically, the optimal global explanation is one that maintains the weighted ranking order in explanations over all coalitions. We use a sampling based approach [12] to solve this hard combinatorial problem that converges to the optimal ranking asymptotically.

We show through experiments that our approach focuses on relevant explanations better than other approaches such as LIME, SHAP or relational explainers using the full MLN distribution [5]. Further, we also conduct a user study to evaluate the usability of our explanations.

This research was sponsored by NSF IIS award #2008812 and NSF award #1934745. The opinions, findings, and results are solely the authors' and do not reflect those of the funding agencies.

II. Related Work

Guidotti et al. [8] survey different explanation approaches in Machine learning and organize them as outcome explanation methods or model explanation methods. Outcome explanations are used for "black-box" classifiers that are complex to explain. A common theme here is to explain the black-box classifier with a simpler model. LIME [19] explains an instance by perturbing it and learning a local, simple decision boundary to classify the perturbed instances. SHAP [13] generates coalitions of features and quantifies the influence of a feature by minimizing the loss between a simple classifier on the coalitions and the original model. Koh and Liang [11] use influence functions from robust statistics to measure the change in training parameters due to a small change in the data. Fong and Veladi [7] use interpretable image perturbations to recognize salient image features. Suderrajan et al. [23] developed integrated gradients as an approach to explain deep networks more generally. More recently, Shao et al. [21] developed an approach using influence functions to correct the model during training such that it gives better quality explanations. In symbolic AI models, Darwiche and Hirth [2] developed a formal framework for explaining classifier decisions using ordered binary decision diagrams compiled from a Bayesian network. Shih et al. [22] explained Bayesian networks by compiling them into decision trees. Farabi et al. [5] explained MLN formulas but unlike our approach they generate explanations using the full MLN which can be non-interpretable if the MLNs represent large, complex distributions. More recently, Broeck et al. [3] analyzed the complexity of SHAP and showed that it is intractable even for simple distributions.

III. Interpretable Explanations

We motivate our approach with a simple example. Consider an MLN with three formulas, $\texttt{Smokes}(x) \Rightarrow \texttt{Asthma}(x)$; $\texttt{Smokes}(y) \wedge \texttt{Friends}(x,y) \Rightarrow \texttt{Asthma}(x)$ and $\texttt{Smokes}(x) \Rightarrow \texttt{Smokes}(x)$. Generally speaking, the first formula is a good explanation for an individual having asthma. However, for an individual, say $Alice$ who does not smoke, $\texttt{Asthma}(Alice)$ is explainable if a lot of Alice's friends are smokers. Thus, the relational dependencies between $Alice$ and other individuals is an effective explainer for $Alice$ having asthma. However, suppose $Alice$ is friends with a large number of individuals, it becomes harder to quantify the influence of each of these individuals since their smoking habits depend upon other individuals who are friends with them. In general, as the domain (number of individuals or objects) become larger, the probabilistic graphical model underlying the MLN becomes complex and extracting the important dependencies for a specific query becomes harder.

A common strategy that is used in well-known explanation methods for black-box classifiers is to estimate the hard-to-interpret black-box classifier boundary using simpler *surrogate* functions. For instance, SHAP creates *feature coalitions* that combines subsets of features and learns a surrogate model from these coalitions from which the explanation is derived. Similarly, LIME perturbs instances in the neighborhood of the predicted instance and learns a surrogate model for explaining the classification of the perturbed instances in the local-neighborhood. Following the same principle, here, we simplify the MLN creating smaller coalitions of formulas. We rank the importance of formulas within each coalition and then combine these explanations together. Since each of the explanations are extracted from a simpler MLN, the inference results on the model are more reliable and the explanations for the inference results are likely to be more interpretable.

A. Explanation Framework

To formalize our explanation framework, we begin with some terms and notation. An MLN formula combines predicates using logical connectives.We assume that all formulas are universally quantified clauses. A *predicate* has multiple arguments. A variable (x) representing an argument in a predicate has can substituted by a finite *domain* (Δ_x) (Herbrand semantics). A *ground atom* is a predicate where all its variables have been substituted by *objects* from their corresponding domains. A *ground formula* only consists of ground atoms. Each ground atom is a binary random variable ($\texttt{True}$ / $\texttt{False}$ or 0/1) and each ground formula is a potential function in the Markov network representation of an MLN. An assignment to all the ground atoms of the MLN is called a possible *world*.

Let $\mathcal{M}$ represent the MLN and $P()$ represent the distribution of the MLN. Let $\mathbf{f}(Q)$ represent the set of ground formulas containing the query atom Q and let $\mathbf{E}$ represent the set of evidence atoms. Note that here, we assume that each atom is either an evidence or a query atom. Let $\mathbf{Q}_{-Q}$ represent the set of possible assignments to all query atoms except $Q \in \mathbf{Q}$. Let w_f represent the weight of a ground formula f (note that all groundings of a first-order formula share the same weight in the regular MLN semantics). The marginal probability is defined as follows.

$$P(Q|\mathbf{E}) = \sum_{Q' \in \mathbf{Q}_{-Q}} \frac{1}{Z} \exp\left(\sum_{f} w_f n_f\right) \tag{1}$$

where Z is the normalization constant of $\mathcal{M}$. Specifically, we need to sum out $\mathbf{Q}_{-Q}$ to compute the marginal for Q. We define an explanation for query Q as follows.

Definition 1. *The explanation for query Q in $\mathcal{M}$ with evidence $\mathbf{E}$ is denoted by $\sigma(Q)$ is a permutation π over $\mathbf{f}(Q)$.*

Given an importance weighting function $I()$, where $I(f, Q)$ indicates the importance of ground formula f on determining the probability of the query $P(Q)$, a sound explanation is defined as follows.

Definition 2. *$\sigma(Q)$ is a sound explanation if π orders $f_i \in \mathbf{f}(Q)$ such that $I(f_{\pi_i}, Q) \geq I(f_{\pi_{i+1}}, Q)$.*

In the remainder of the paper, when we refer to an explanation given an importance function, we assume that it is a sound explanation. To define $I()$, an intuitive approach is to directly use the MLN parameters. Specifically, $I(f, Q)$

$= w_f$, where w_f is the weight of f. However, this simple approach is problematic since the importance of a formula changes dynamically based on the observed variables $\mathbf{E}$. For example, let $\texttt{Fever}(x) \Rightarrow \texttt{Flu}(x)$ have a larger weight compared to $\texttt{Cough}(x) \Rightarrow \texttt{Flu}(x)$. However, given evidence that an individual has cough and not fever, we can explain that the individual has flu using the second formula. Similarly for a different query, the first formula may be more important than the second. Thus, the underlying problem with assuming that the MLN weights determine the importance of a formula is that the parameters do not correspond to probabilities. For the explanation to be meaningful, given evidence $\mathbf{E}$, $I()$ should encode the influence of a formula on the conditional probability $P(Q_1, \ldots Q_n|\mathbf{E})$ where $Q_1, \ldots Q_n$ represent the query atoms. We define this as follows.

Definition 3. *Let $\mathbb{E}[n_f]$ be the expected truth value of formula f w.r.t the joint conditional distribution $P(Q_1, \ldots Q_n|\mathbf{E})$ and $\mathbb{E}_Q[n_f]$ be the expected truth value of f w.r.t the marginal distribution of Q, i.e., $P(Q|\mathbf{E})$, we define the importance weight of formula f for a query Q as*

$$I(f, Q) = \mathbb{E}_Q[n_f] - \mathbb{E}[n_f] \tag{2}$$

Intuitively, using the above definition, $I(f, Q)$ has a larger value if f is true in more worlds where Q is also true and false in more worlds where Q is false. More specifically, we relate the importance $I(f, Q)$ to the partial derivative of the marginal probability w.r.t to the weight of f.

Proposition 1. $I(f, Q) = \frac{\partial \log(P(Q|\mathbf{E})}{\partial w_f}$

Proof. Taking logs in Eq. (1) we have,

$$logP(Q|\mathbf{E}) = log \sum_{Q' \in \mathbf{Q}_{-Q}} \exp(\sum_f w_f n_f) - log(Z)$$

Taking partial derivatives we have,

$$\frac{\partial logP(Q|\mathbf{E})}{\partial w_f} = \frac{1}{Z'}\frac{\partial Z'}{\partial w_f} - \frac{1}{Z}\frac{\partial Z}{\partial w_f}$$

where $Z' = \sum_{Q' \in \mathbf{Q}_{-Q}} \exp(\sum_f w_f n_f)$. Simplifying terms we get,

$$\sum_{Q' \in \mathbf{Q}_{-Q}} P(Q, Q'|\mathbf{E}) n_f - \sum_{\bar{Q}, Q' \in \mathbf{Q}_{-Q}} P(\bar{Q}, Q'|\mathbf{E}) n_f$$

where $\bar{Q}$ denotes a possible assignment to Q. Thus,

$$\frac{\partial logP(Q|\mathbf{E})}{\partial w_f} = \mathbb{E}_Q[n_f] - \mathbb{E}[n_f]$$

□

From the above theorem, the importance $I(f, Q)$ is proportional to the influence of f on $P(Q|\mathbf{E})$. Note that the importance of a formula is dynamic (as compared to the MLN weight) since it depends on the query as well as the evidence.

B. Sampling-based Importance Estimation

From Eq. (2), to compute $I(f, Q)$, we need to compute expectations w.r.t $P(Q|\mathbf{E})$ and $P(Q_1, \ldots Q_n|\mathbf{E})$. Therefore, computing $I(f, Q)$ is computationally infeasible since computing the expectations is equivalent to solving the marginal inference problem which is well-known to be in $\#P$. Therefore, we use a sampling-based approach to approximate $I(f, Q)$.

Specifically, we use Gibbs sampling (though in theory other samplers can be utilized) due to its efficiency to estimate $I(f, Q)$. Specifically, we generate samples from the distribution and approximate the expected values $\mathbb{E}_Q[n_f]$ and $\mathbb{E}[n_f]$. To implement this, we start with a random configuration of $\mathbf{Q}$. In each iteration, we sample a random query Q from its conditional distribution $P(Q|\mathbf{Q}_{-Q})$. If the sampled value is $Q = 1$ (or `True`), then for all `True` formulas that contain Q, we update the estimate for $\mathbb{E}_Q[n_f]$ and for all `True` formulas, we update the estimate for $\mathbb{E}[n_f]$.

Algorithm 1: Explanations

```
   Input: MLN $\mathcal{M}$, evidence $\mathbf{E}$, queries $\mathbf{Q}$
   Output: Explanations for $\mathbf{Q}$
   //
1  $\bar{\mathbf{Q}}$ = random assignment to $\mathbf{Q}$
2  for $t = 1$ to $T$ do
3      for Each atom $Q \in \mathbf{Q}$ do
4          Sample $Q$ from $P(Q|Q_{-Q}, \mathbf{E})$ and update $\bar{\mathbf{Q}}$
5          for each $f$ containing $Q$ do
               // update sufficient statistics
                   for $\mathbb{E}[n_f]$
6              Update the count $n_1(f, Q)$ if $f$ is true
               // update sufficient statistics
                   for $\mathbb{E}_Q[n_f]$
7              if $Q$ is sampled as true then
8                  Update the count for $n_2(f, Q)$ if $f$ is true
9      for each $Q$ in $\mathbf{Q}$ do
10         for each $f$ containing $Q$ do
11             $\bar{I}(f, Q) = \frac{1}{t}\ (n_2(f, Q) - n_1(f, Q))$
```

Algorithm 1 summarizes our sampling-based approach for explanations. Based on the convergence of the probabilities for the Gibbs sampler in Algorithm 1, clearly, the expected values in Eq. (2) converge to the true expected values. Therefore, as $T \rightarrow \infty$, $\bar{I}(f, Q) \rightarrow I(f, Q)$.

C. Influence of Domain-Size

The *mixing time* of the Gibbs sampler in Algorithm 1 is the time that the sampler takes to reach the stationary distribution, namely, the distribution $P()$. $I(f, Q)$ is estimated from samples after the sampler reaches its stationary distribution, i.e., after a period called its *burn-in* time. If $I(f, Q)$ is estimated from samples before burn-in, the explanations will be unreliable since they are not consistent with the distribution represented by the MLN. However, the main issue is that for large MLNs, the mixing time can be extremely large and consequently, the explanations generated from the samples are likely to be of non-interpretable.

Intuitively, when the MLN structure is more complex, the distribution becomes harder to sample and explain. This can be formalized by results in Sa et al. [20]. Specifically, the mixing time depends upon the *total influence* in the MLN distribution. To define this, let $\mathbf{B}_j$ denote the set of all pairs of states ($\bar{X}$, $\bar{Y}$). Each each state is an assignment to all atoms, and each pair ($\bar{X}$, $\bar{Y}$) $\in \mathbf{B}_j$ differ in an assignment to a single variable j. The total influence is the maximum total variational distance between distributions obtained by conditioning an atom on the assignments in the pairs defined in $\mathbf{B}_j$. Formally,

$$\alpha = \max_{V \in \mathcal{M}} \sum_j \max_{(\bar{X},\bar{Y}) \in \mathbf{B}_j} ||P(V|\bar{X}_{-V}) - P(V|\bar{Y}_{-V})||_{TV} \quad (3)$$

where $P(V|\bar{X}_{-V})$ is the conditional distribution of an atom $V \in \mathcal{M}$ given assignments to all the other atoms.

For larger values of α in Eq.(3), in [20] it is shown that the Gibbs sampler takes longer to mix. Thus, using Algorithm 1, it becomes harder to generate meaningful explanations. Specifically, in our case, the difference between $P(V|\bar{X}_{-V})$ and $P(V|\bar{Y}_{-V})$ is dependent upon the influence of a single atom. Specifically, if the change to an atom's assignment modifies the truth values of several ground formulas, then the total variational distance in Eq. (3) increases.

Formally, let $C(f, R)$ represent the variables in formula f that are part of predicate R. Let $C(f, R)^{-}$ represent variables in formula f that are not part of predicate R. Let ($\bar{X}$, $\bar{Y}$) $\in \mathbf{B}_j$ and j denote an atom of predicate type R. For any atom V, the variational distance between $P(V|\bar{X}_{-V})$ and $P(V|\bar{Y}_{-V})$ is proportional to the number of groundings of f whose truth value (either 0/1) differ in the two distributions. The atom j occurs in $\prod_{v' \in C(f,R)^{-}} |\Delta_{v'}|$ ground formulas, where $\Delta_{v'}$ is the domain of variable v'. Therefore, a change to the state of j can potentially change the truth value of all these ground formulas. Thus, as the domains of the variables increase, the variational distance between the state pairs in $\mathbf{B}_j$ increases and the total influence becomes larger. Algorithm 1 therefore generates non-interpretable explanations in MLNs with large domains. To generate interpretable explanations, we simplify the MLN such that the total influence in the simplified MLN is smaller than that in the original MLN.

D. Relational Coalitions

To reduce the total influence, we create *coalitions* of ground formulas that are much smaller than the full MLN. On such coalitions, the sampler in Algorithm 1 mixes faster and is more likely to generate interpretable explanations. In principle, this approach is similar to LIME and SHAP where the original classifier is approximated by a simpler, more explainable classifier. One way to generate coalitions is to sample ground formulas in $\mathbf{f}(Q)$ randomly for each query Q. However, since the MLN is a relational model a randomly sampled coalition may not preserve the relational dependencies present in the original MLN. To illustrate this, consider the graph underlying an MLN shown in Fig. 1.Suppose we randomly sample this graph, we may end up with coalitions shown in Fig. 1 (b). However, this does not truly capture all dependencies in the original graph. Instead, if we sample the graph to obtain coalitions shown in Fig. 1 (c), though this model is simpler, we still retain the dependencies specified in the original graph. More generally, by exploiting symmetries in the MLN, we create coalitions that preserve relational dependencies.

Definition 4. *Given evidence* $\mathbf{E}$*,* X_1 *and* X_2 *are exchangeable objects if* X_1 *and* X_2 *can be exchanged in the ground formulas such that the formulas which were satisfied (or unsatisfied) by* $\mathbf{E}$ *before the exchange remain satisfied (or unsatisfied) after the exchange.*

Two coalitions $\mathbf{L}_1$ and $\mathbf{L}_2$ are symmetric if for every formula in $\mathbf{L}_1$ can be uniquely mapped to a formula in $\mathbf{L}_2$ such that the objects in $\mathbf{L}_1$ are exchangeable with objects in $\mathbf{L}_2$. For example, the coalition, $\mathtt{R}(X_1) \wedge \mathtt{S}(X_1, Y_1)$; $\mathtt{R}(X_1) \wedge \mathtt{S}(X_1, Y2)$ is symmetric to $\mathtt{R}(X_2) \wedge \mathtt{S}(X_2, Y_1)$; $\mathtt{R}(X_2) \wedge \mathtt{S}(X_2, Y2)$ if X_1 is exchangeable with X_2. Suppose we are given equivalent coalitions, $\mathbf{L}_1$ and $\mathbf{L}_2$, where a ground formula where $f \in \mathbf{L}_1$ is mapped to $f' \in \mathbf{L}_2$, then, if f is an explanation to a query w.r.t $\mathbf{L}_1$, we project f' as the explanation to Q w.r.t $\mathbf{L}_2$.

While truly exchangeable objects may be rare in practice, we can soften the constraints in our definition to allow for approximately exchangeable objects. Specifically, we define a continuous approximation for the exchangeability between X_1 and X_2, $\delta(X_1, X_2)$ based on the number of ground formulas whose assignments differ before and after the exchange of X_1 and X_2. However, computing this for large MLNs is hard and it is shown in prior work [24] that counting the satisfied groundings of a formula given evidence is a computationally hard problem. Therefore, it is infeasible to exactly compute $\delta(X_1, X_2)$. However, we instead leverage scalable learning methods to learn a dense vector representation that approximately encodes how similar X_1 is to X_2. Specifically, as in [10], we learn embeddings over objects in the MLN using an approach commonly used in word embedding methods. Specifically, suppose a ground formula f that is satisfied by the evidence contains objects ($X_1 \ldots X_k$), we predict X_i from $X_1 \ldots X_{i-1}$, $X_{i+1} \ldots X_k$. Thus, if X and X' are predicted from similar objects (or have the same context), this means that X and X' can be exchanged without significantly altering the truth values of the formulas in which they occur. To learn the embedding, we use existing implementations of skip-gram models [15]. Note that several other graph-based methods such such as those computing automorphisms in the MLN graphs to recognize symmetries [1, 17], locality-sensitive hashing [16], etc. can be used as well. However, it is easy to see that, to compute the embedding in our case, we do not explicitly construct the MLN graph which can be very large, and therefore we can scale up even to large MLNs.

Given the embeddings, we cluster the objects to generate coalitions such that for a query, we choose formulas in the coalition that can effectively substitute for formulas that are not chosen in the coalition.

Definition 5. *Given a clustering of objects based on their*

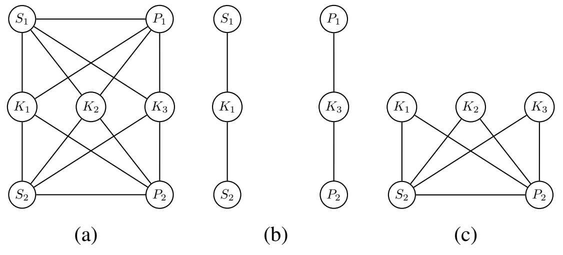


Fig. 1: (a) Original MLN Graph. (b) Simplification by random sampling. (c) Simplification that preserves relational structure.

embeddings, if f is a formula with objects ($O_1 \ldots O_k$), $\theta(f)$ is an approximately symmetric formula where each O_i is substituted by O'_i that is in the same cluster as O_i.

We construct a coalition by selecting formulas based on a clustering of objects. Specifically, we sample objects from each cluster such that the number of objects sampled is proportional to the cluster size. Let $\mathbf{O} = O_1 \ldots O_m$ be the sampled objects. For every query atom that can be formed from the sampled objects, say Q, we construct the coalition $\mathbf{f}'(Q)$ as follows. Initially, we start with an empty $\mathbf{f}'(Q)$. For each ground formula f that contains Q, we include f in $\mathbf{f}'(Q)$ if we cannot find $\theta(f) \in \mathbf{f}'(Q)$. Thus, we reduce the original set of formulas $\mathbf{f}(Q)$ to $\mathbf{f}'(Q)$ such that for $f \in \mathbf{f}(Q) \setminus \mathbf{f}'(Q)$, there exists a $\theta(f) \in \mathbf{f}'(Q)$. This means that, suppose the optimal explanation for Q is a ground formula f, then the coalition can potentially generate f or $\theta(f)$ as its explanation.

E. Integrating Multiple Explanations

By combining coalitions across all sampled queries, we generate an MLN and jointly explain all the sampled queries using Algorithm 1. However, note that since the embeddings and the clustering is approximate, using a single coalition, we may not obtain the formulas needed to explain a query effectively. Therefore, we generate multiple coalitions and explain each independently. We then integrate the explanations generated across all the coalitions.

Let $\sigma_i(Q)$ represent the explanation obtained for query Q using $\mathcal{M}_i$ which is the MLN generated in the i-th iteration of sampling the clusters. Note that since we are sampling the clusters, not every query will be a part of each of the generated MLNs. Therefore, if a query $Q \notin \mathcal{M}_i$ we can generate an explanation for Q using $Q' \in \mathcal{M}_i$ that is in the neighborhood of Q in the embedding assuming that the two explanations are symmetric. Specifically, if f is a formula in the explanation for Q', we explain Q with a substitution $\theta(f)$ that contains Q.

Given explanations $\sigma_1(Q) \ldots \sigma_m(Q)$, where each explanation is from a coalition that tries to cover the full relational structure of the MLN, we now generate a unified explanation for Q. Specifically, we want to know the influence of a formula f on a query across the coalitions. In the case of non-relational models, note that predictions are typically easy to perform. Therefore, for each coalition, it is relatively simple to optimize the loss between the prediction by the original model for that coalition instance with the predication made by the approximate, simpler model (e.g. a linear model) to derive the final explanation across coalitions. However, in the case of relational models, inference is hard and therefore, we cannot assume that we can perform inference in the original model in the first place. That is, generating results based on the full MLN is a hard problem when the MLN is large even using approximate inference methods.Therefore, instead of optimizing the loss between inference results from the original MLN with inference results from the coalitions, we derive weighted explanations, where the weight approximately encodes the difference between the original MLN and the simpler MLN. The weighted explanations are then combined into a unified global explanation for the model.

1) Coalition Weighting: Let β_i denote the weight for the explanations derived from $\mathcal{M}_i$. Note that, ideally, we want to weight $\sigma_i(Q)$ based on the distance between $\mathcal{M}_i$ and $\mathcal{M}$. However, this is infeasible since the exact distribution of $\mathcal{M}$ is intractable to compute. Therefore, we instead weight each coalition based on how the importance weights computed from the coalition match with the MLN parameterization. Influence functions proposed in [11] use a similar approach where the change in model parameters relative to perturbed parameters is used as a way to quantify the effect of the perturbation.

Formally, let $\mathcal{M}$ be the original MLN and $\mathcal{M}_i$ be the MLN that was generated from the coalitions. Let $\mathbf{w}$ be the weights (or parameters) of $\mathcal{M}$ and let $\mathbf{w}_i^*$ be the optimal parameterization for $\mathcal{M}_i$. This means, if $\mathbf{w}_i^*$ was used to parameterize $\mathcal{M}_i$, then the importance weights for explanations from the MLN would ideally match with its formula weights. Suppose we are explaining query Q_j using $\mathcal{M}_i$, we compute the importance weights according to Eq. (2). In Proposition 1, we show that the importance weights computed for Q_j is equivalent to the gradient vector for the likelihood function of Q_j. Suppose this gradient has a large norm, this means that to the importance weights are significantly different from the weights $\mathbf{w}$. That is a large gradient norm implies that $\mathbf{w}$ must be significantly changed to reach $\mathbf{w}_i^*$. Thus, the explanations generated by $\mathcal{M}_i$ have a larger bias and should therefore be weighted down relative to the other generated simple MLNs. On the other hand, if the importance weights imply a small gradient norm, then the difference between $\mathbf{w}$ and $\mathbf{w}_i^*$ is small.

In general, we can now weight the coalitions based on the importance weights of the formulas generated while explaining the queries in the coalition. That is, if the importance weights of the formulas show a large variation over all the queries, then those coalitions will have a smaller weight compared to coalitions where the importance weights have small variation over all the queries. Specifically, let $\mathbf{I}_i$ represent the matrix of importance weights obtained from $\mathcal{M}_i$, where the j-th row corresponds to weights for query Q_j. Let $z_i = ||\mathbf{I}_i||_F$, where $||\mathbf{I}_i||_F$ denotes the Frobenius norm for the weight matrix. Let $\bar{Z} = \sum_i z_i$. We weight the explanations from $\mathcal{M}_i$ with $\beta_i = 1 - \frac{z_i}{\bar{Z}}$.

2) Unified Explanation Ranking: Let $S(Q) = \cup_i \sigma_i(Q)$, i.e., the union of explanations from all MLNs. To make equations more readable, we drop the Q since it is implicit that explanations are specific to a query. Let τ represent an ordered (or ranked) subset of k explanations, i.e., $\tau \subset S$. We now define a distance function between τ and each of the explanations, where the explanation for $\mathcal{M}_i$ is weighted by β_i as,

$$\Phi(\tau) = \sum_i \beta_i d(\tau, \sigma_i) \tag{4}$$

where $d(\tau, \sigma_i) = \sum_{t \in \tau \cup \sigma_i} |R_i(t) - R_\tau(t)|$, $R_i(t)$ is t's ranking in σ_i and $R_\tau(t)$ is its ranking in τ. Note that if t is not in τ, we set $R_\tau(t)$ to the maximum value $k+1$, and similarly if t is not in σ_i, we set $R_i(t)$ to $k+1$. We formulate the optimal explanation as,

$$\tau^* = \arg\min_\tau \Phi(\tau)$$

Solving the optimization problem for τ^* exactly is computationally hard since we need to enumerate all possible subsets of size k. Therefore, we use an approximate approach to combine the weighted explanations using the Cross-Entropy Monte Carlo (CEMC) [12] algorithm. Specifically, let $\mathbf{v}$ be a $n \times k$ probability matrix, where n is the total number of formulas in the union of all explanations and k is the size of the global explanation that we seek to find. Each column in $\mathbf{v}$ represents a multinomial distribution over the formulas. To obtain a global explanation, we can draw a sample from the distribution $P_\mathbf{v}$ to generate $\mathbf{x}$ such that each column in $\mathbf{x}$ has exactly a single 1 and each row sums to at most 1. Given that $\mathbf{v}$ is the current set of parameters, in [12], it is shown that new parameters $\mathbf{v}'$ that minimizes the KL-divergence between the current probability distribution and the ideal distribution is given by maximizing,

$$\mathbb{E}_\mathbf{v}[\mathcal{I}(\Phi(f(\mathbf{x}, \mathbf{v})) \leq y) log P_{\mathbf{v}'}(\mathbf{x})]$$

where $\mathcal{I}$ is an indicator function and $f(\mathbf{x}, \mathbf{v})$ denotes a τ that has been drawn from $P_\mathbf{v}$. To do this, we draw samples from $P_\mathbf{v}$ and count the proportion of samples for which the objective function value is smaller than a given y. Specifically,

$$v_{new} = \frac{\sum_{i=1}^N \mathcal{I}(\Phi(\tau_i) \leq y)\mathbf{x}_i}{\sum_{i=1}^N \mathcal{I}(\Phi(\tau_i) \leq y)} \tag{5}$$

CEMC learns an approximation for the optimal global explanation as follows. We initialize the probability matrix $\mathbf{v}_o$ as a uniform distribution for each column. In each iteration, we consider N samples, where each sample is a possible explanation. Half the samples are drawn from $\mathbf{v}_i$ and we augment it with half of the best samples (best objective function values) from $\mathbf{v}_{i-1}$. We then order the explanations in ascending order of their objective values computed using Eq. (4). Suppose Φ_1 ... Φ_N are the ordered objective values for the explanations, we choose y to be the ρ-quantile (for a suitable rho) objective value. Using this, we obtain the updated $\mathbf{v}_{i+1}$ from Eq. (5). After the parameters in the probability matrix converge, we output the final explanation as the one that corresponds to the best objective function value, i.e., $y = \Phi_1$. CEMC generates asymptotically unbiased explanations, i.e., Φ_1 converges to the optimal objective value in the limit.

IV. Experiments

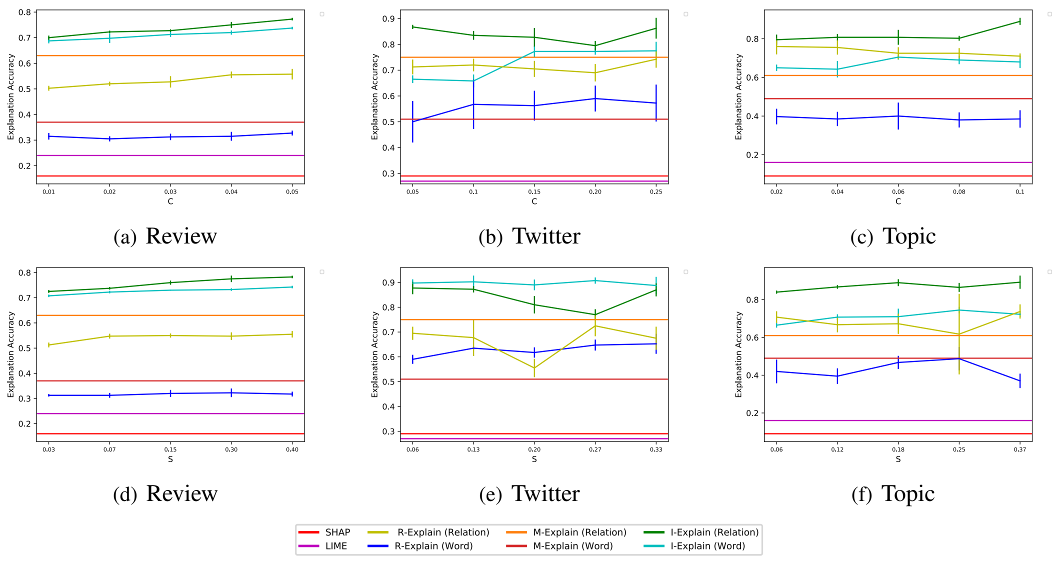
We evaluate our approach along three dimensions, i) we evaluate accuracy of explanations based on manual annotations, ii) we evaluate the importance of information present within the explanations and iii) we evaluate the usability of explanations through a user-study.

A. Data and Tasks

We use three datasets in our evaluation. The first dataset is a dataset sampled from Yelp [18] for review sentiment classification that contains 1544 reviews. Our second dataset consists of 650 COVID tweets from Kaggle. We classify whether a COVID-19 tweet contains useful information or not. We manually annotated tweets based on whether they contain facts that can be considered as useful information (e.g. scientific details, policies, etc.) or if the tweets are non-informative. Our third dataset uses a portion of the well-known 20newsgroups dataset from the UCI repository. Here, we classify articles as automative related articles or otherwise. We had a balanced dataset of 1000 articles. The MLNs in each case consist of word formulas, i.e., connecting words with the query and relational formulas that encode homophily. That is, if two queries are linked then they share the same class. In reviews, the links are defined by reviews written by same user and those about the same restaurant. In the twitter and topics data, the links are defined by tweets (or topics) written by the same user. Note that, for more complex relational formula structures, in general, sampling becomes harder [24]. We will explore approximations for such formulas in future.

B. Implementation

We refer to our approach as `I-Explain`. To learn the embeddings for MLN objects, we use open source code from [10]. We learn the MLN weights using a hybrid approach where we initialize the word formula weights using SVM coefficients and then learn the relational formula weights conditioned word weights [6] using Max-likelihood estimation (as in standard MLN learning). We did not use hard constraint formulas since Gibbs sampling tends to work poorly here. In future, we will look to extend our explanations to samplers (e.g. slice samplers) that can better handle determinism. We used an open source R package for CEMC (with default parameters) to combine the coalition explanations. We applied `LIME` and `SHAP` to our tasks using the word features since they do not generate relational explanations. We also developed a baseline where we created coalitions by randomly sampling formulas (denoted by `R-Explain`. Finally, we also compared our approach with explanations from the complete MLN as described in [5] which we refer to as `M-Explain`. We manually annotated the ground truth for the explanations for all instances in our datasets. Specifically, since all our tasks are text based, for each instance, we pick words that best explain the class. To avoid bias, these were annotated by two people independently and the final annotation was the common explanations chosen by both (if there were no common words,


(a) Review (b) Twitter (c) Topic

(d) Review (e) Twitter (f) Topic

Fig. 2: The mean accuracy values (varying values of C ans S) for 10 runs are plotted along with error bars that indicate Std-Dev.

Dataset	Method	Top-Exp-Acc (F1)	Peak-Acc (F1)	Std-Dev
Reviews	I-Explain	0.82	0.84	0.01
	R-Explain	0.54	0.58	0.04
	M-Explain	0.6	0.68	0.01
	SHAP	0.53	0.58	0.01
	LIME	0.55	0.61	0.02
Tweets	I-Explain	0.79	0.83	0.01
	R-Explain	0.61	0.77	0.08
	M-Explain	0.6	0.69	0.01
	SHAP	0.33	0.44	0.01
	LIME	0.46	0.54	0.01
Topics	I-Explain	0.7	0.75	0.008
	R-Explain	0.58	0.66	0.04
	M-Explain	0.62	0.67	0.01
	SHAP	0.39	0.54	0.02
	LIME	0.53	0.56	0.01

TABLE I: Prediction Accuracy (F1-score) using only the explanation (Top-Exp-Acc) and the peak accuracy obtained when we add the formulas (or features) in ranked order according to explanation importance (Peak-Acc). The mean and std-dev for 10 runs is shown.

we sampled words from both explanations). Annotating relational formula explanations manually is hard. Therefore, for a query Q, for each true grounding of a relational formula f where Q is related to Q', we measure similarity of Q and Q' using Gensim's Doc2Vec. If Q is similar to Q', then f is an explanation to both Q and Q'.[1]

C. Explanation Accuracy

We computed the accuracy as the $\%$ of matches of the generated explanations with the annotated explanations for each dataset. To take into account accuracy of predictions, we only considered explanation matches for correct predictions and for wrong predictions, we automatically assumed the explanation to be wrong. For `I-Explain`, `R-Explain` and `M-Explain`, we used the top 5 ranked word formulas and the top 5 ranked relational formulas as the explanation. Note that for `LIME`, `SHAP`, we only obtain word explanations and therefore only measure accuracy on these. We evaluated our approach by varying two parameters C and S, C controls the number of clusters and S controls the number of samples drawn from each cluster to create the coalitions. For a domain-size of $|\Delta_x|$, we use $C * |\Delta_x|$ clusters and if a cluster contains N instances, we use $S * N$ samples from that cluster. For `R-Explain`, we pick random formulas to create coalitions such that it matches the number of formulas in $\mathtt{I-Explain}$ for a fair comparison. We ran the experiments 10 times and report the mean and variance of the accuracy. Our results are shown in Fig. 2, where we independently show the accuracy on word explanations and relational formula explanations. For `M-Explain`, `LIME` and `SHAP`, clusters don't play any role, so their accuracy is constant. Fig. 2 (a), (c), (e) keeps S constant at 5%, while (b), (d), (f) keeps C constant at 2% for the review and topic data, and 5% for twitter (since it is around half the size) and varies S. From the results, we see that `I-Explain` consistently shows better accuracy than other methods. As C increases, we have larger coalitions and the performance is fairly stable with slightly increasing accuracy but can also dip in some cases (see Fig. 2 (c)). This indicates that with larger coalitions, we may have more uncertainty (since the expected values are estimates), therefore explaining smaller coalitions is an advantage in relational models.

D. Explanation Information Content

If explanations contain content that is more important in the model, then, they should be able to make accurate predictions using only the explanations. Table 1 summarizes the accuracy in terms of F1-score when we only add the explanations for each query and use them to make predictions. Further, we also show the peak accuracy that is achieved by the model as

[1]Data, annotations and code is available at https://github.com/khanfarabi/IEEE-Big-Data-2021

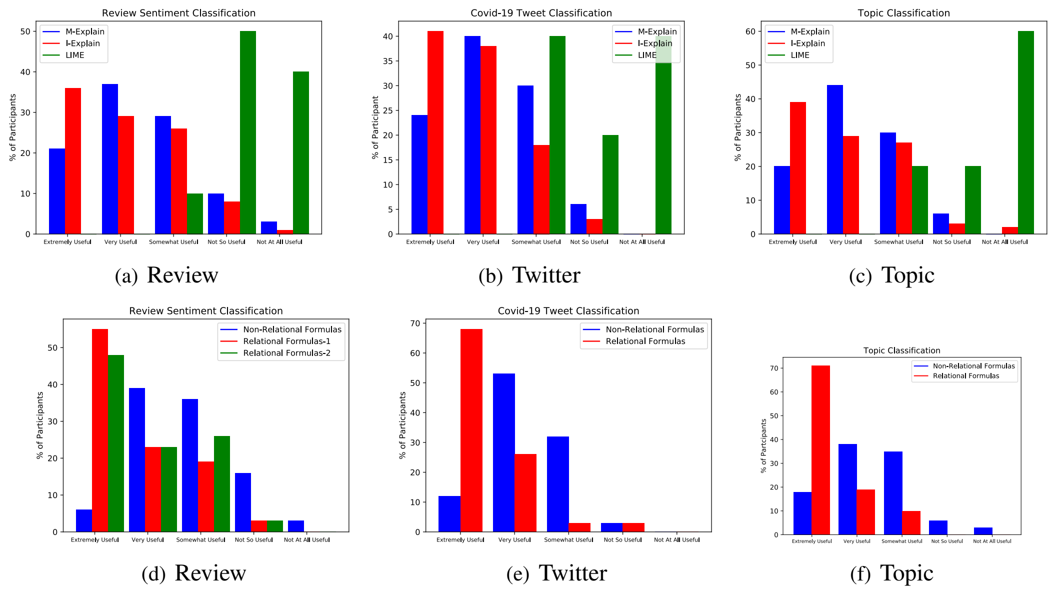


Fig. 3: (a) - (c) The y-axis shows the $\%$ of participants who rated the explanation feature with the rating specified in the x-axis where the $\%$ is normalized by the number of features in the explanation. (d) - (f) User ratings for relational and non-relational features for `I-Explain`.

we add formulas (or features in the case of LIME or SHAP) in order of explanation importance. We show the results in Table I where we indicate the mean F1-score for the predicted queries for 10 runs as well as the standard deviation. As seen here, `I-Explain` has better accuracy when we only use the explanation for the prediction. Also, the peak accuracy is close to the accuracy using explanations in most cases indicating that explanations represent highly informative content.

E. *Usability*

We evaluate the usability of explanations through a user study. Our goal here is to understand if users find the explanations useful. We recruited 50 graduate students in this study who were currently (or formerly) enrolled in a Machine Learning course. Note that this demographic is likely to constitute potential XAI users since one of the key applications of XAI is to help experts debug Machine learning methods. For more naive users, it is our opinion that we may require more simplified explanation interfaces which we hope to explore in future. We divided the overall group into 3 sub-groups and a student was given explanations from just one of the methods (which was not named) to avoid bias. To ensure sufficient responses for each method, we used `I-Explain`, `M-Explain` and `LIME` (since `SHAP` explanations are similar in format to `LIME`).

The explanation dashboard consists of non-relational explanations (words) and relational explanations. Visualizing relational formula explanations is not as straightforward since relationships are not explicitly seen in the data (as opposed to words). To visualize this, we average the importance weights of the relational formulas in the explanation and display this as a graph (normalized between 0 and 1) indicating support of the relational formulas in the prediction. Our survey consisted of 12 randomly chosen explanations where we used 4 explanations for each task (2 of them corresponding to each class). For every explanation, we asked users to rate the usefulness of each the word and relational explanations in the dashboard on a 5-point Likert scale. The results of the survey are shown in Fig. 3. Fig. 3 (a) - (c) show the $\%$ of user responses for each value in the Likert scale. We see here that `I-Explain` and `M-Explain` had better scores than `LIME`. This shows that overall, users preferred to see explanations with both relational and non-relational formulas. Further, users also preferred `I-Explain` to `M-Explain` since the explanation quality was better due to the use of simplified models. That is, `M-Explain` tends to provide poor explanations when the MLN is large. Fig. 3 (d) - (f) further shows the breakdown of scores for `I-Explain`. Interestingly, across all datasets while users felt both types of features are important, a larger percentage found relational features to be extremely useful in the explanations compared to non-relational features. The mean user score for the relational features in the explanation was 4.42 while for the non-relational features, it was 3.84. A two-sided t-test showed that the difference in user responses for relational and non-relational features was statistically significant.

V. Conclusion

Explaining the results of marginal probabilistic inference in an interpretable manner is hard when MLNs have large domains. We presented an approach that constructs simplified models from the MLN to generate more interpretable explanations. The simplified explanations are then weighted and combined into a unified explanation. Our results on several problems illustrated that our approach generates high quality explanations for relational data.

References

[1] Hung Hai Bui, Tuyen N. Huynh, and Sebastian Riedel. Automorphism groups of graphical models and lifted variational inference. In *UAI*, 2013.

[2] Adnan Darwiche and Auguste Hirth. On the reasons behind decisions. *CoRR*, abs/2002.09284, 2020.

[3] Guy Van den Broeck, Anton Lykov, Maximilian Schleich, and Dan Suciu. On the tractability of SHAP explanations. In *AAAI*, 2021.

[4] P. Domingos and D. Lowd. *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan & Claypool, San Rafael, CA, 2009.

[5] Khan Mohammad Al Farabi, Somdeb Sarkhel, Sanorita Dey, and Deepak Venugopal. Fine-grained explanations using markov logic. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD*, 2019.

[6] Mohammad Khan Al Farabi, Somdeb Sarkhel, and Deepak Venugopal. Efficient weight learning in high-dimensional untied mlns. In *The 21st International Conference on Artificial Intelligence and Statistics, AISTATS 2018*, pages 1637–1645, 2018.

[7] Ruth C. Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *ICCV*, pages 3449–3457. IEEE Computer Society, 2017.

[8] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM Comput. Surv.*, 51(5):93:1–93:42, 2018.

[9] David Gunning. Darpa's explainable artificial intelligence (XAI) program. In *ACM Conference on Intelligent User Interfaces*, 2019.

[10] Mohammad Maminur Islam, Somdeb Sarkhel, and Deepak Venugopal. On lifted inference using neural embeddings. In *AAAI Conference on Aritificial Intelligence*, pages 7916–7923, 2019.

[11] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *ICML*, pages 1885–1894, 2017.

[12] Shili Lin and Jie Ding. Integration of ranked lists via cross entropy monte carlo with applications to mrna and microrna studies. *Biometrics*, 65:9–18, 2009.

[13] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[14] Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M. Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable ai for trees. *Nature Machine Intelligence*, 2(1):56–67, 2020.

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Neural Information Processing Systems*, pages 3111–3119. 2013.

[16] M. Mladenov, A. Globerson, and K. Kersting. Efficient Lifting of MAP LP Relaxations Using k-Locality. *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, 2014.

[17] Mathias Niepert. Markov chains on orbits of permutation groups. In *UAI*, pages 624–633. AUAI Press, 2012.

[18] Shebuti Rayana and Leman Akoglu. Yelp Dataset for Anomalous Reviews. Technical report, Stony Brook University, 2015. http://odds.cs.stonybrook.edu.

[19] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Knowledge Discovery and Data Mining (KDD)*, 2016.

[20] Christopher De Sa, Ce Zhang, Kunle Olukotun, and Christopher Ré. Rapidly mixing gibbs sampling for a class of factor graphs using hierarchy width. In *Advances in Neural Information Processing Systems 28*, pages 3097–3105, 2015.

[21] Xiaoting Shao, Arseny Skryagin, Wolfgang Stammer, Patrick Schramowski, and Kristian Kersting. Right for better reasons: Training differentiable models by constraining their influence functions. In *AAAI*, pages 9533–9540, 2021.

[22] Andy Shih, Arthur Choi, and Adnan Darwiche. A symbolic approach to explaining bayesian network classifiers. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pages 5103–5111, 2018.

[23] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, pages 3319–3328, 2017.

[24] Deepak Venugopal, Somdeb Sarkhel, and Vibhav Gogate. Just count the satisfied groundings: Scalable local-search and sampling based inference in mlns. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.