information

Article

Numerical Markov Logic Network: A Scalable Probabilistic Framework for Hybrid Knowledge Inference

Ping Zhong *, Zhanhuai Li, Qun Chen, Boyi Hou and Murtadha Ahmed

School of Computer Science, Northwestern Polytechnical University, 127 West Youyi Road, Xi'an 710129, China; lizhh@nwpu.edu.cn (Z.L.); chenbenben@nwpu.edu.cn (Q.C.); ntoskrnl@mail.nwpu.edu.cn (B.H.); a.murtadha@mail.nwpu.edu.cn (M.A.)

* Correspondence: zhongping@mail.nwpu.edu.cn

Abstract: In recent years, the Markov Logic Network (MLN) has emerged as a powerful tool for knowledge-based inference due to its ability to combine first-order logic inference and probabilistic reasoning. Unfortunately, current MLN solutions cannot efficiently support knowledge inference involving arithmetic expressions, which is required to model the interaction between logic relations and numerical values in many real applications. In this paper, we propose a probabilistic inference framework, called the Numerical Markov Logic Network (NMLN), to enable efficient inference of hybrid knowledge involving both logic and arithmetic expressions. We first introduce the hybrid knowledge rules, then define an inference model, and finally, present a technique based on convex optimization for efficient inference. Built on decomposable exp-loss function, the proposed inference model can process hybrid knowledge rules more effectively and efficiently than the existing MLN approaches. Finally, we empirically evaluate the performance of the proposed approach on real data. Our experiments show that compared to the state-of-the-art MLN solution, it can achieve better prediction accuracy while significantly reducing inference time.

Keywords: Markov Logic Network; MAX-SAT inference; loss function decomposition; convex optimization; numeric knowledge rule

Citation: Zhong, P.; Li, Z.; Chen, Q.; Hou, B.; Ahmed, M. Numerical Markov Logic Network: A Scalable Probabilistic Framework for Hybrid Knowledge Inference. *Information* **2021**, *12*, 124. https://doi.org/10.3390/info12030124

Academic Editor: Willy Susilo

Received: 26 January 2021
Accepted: 11 March 2021
Published: 15 March 2021

1. Introduction

In recent years, the Markov Logic Network (MLN) [1] has emerged as a powerful tool for knowledge-based inference due to its ability to combine first-order logic inference and probabilistic reasoning. It has been applied in a wide variety of applications, e.g., knowledge base construction [2–5] and entity resolution [6]. The state-of-the-art probabilistic knowledge-based systems (e.g., Tuffy [7], ProKB [8], and Deepdive [9]) tackle the problem of MLN inference in two steps, grounding and inference. The step of grounding constructs a Markov network by knowledge rules; it is followed by the step of inference, which searches for the Maximum A Posteriori (MAP) probability or marginal probability of the variables.

In many real scenarios, for instance the inference on phone performance as shown in Table 1, knowledge rules may involve both first-order logic and arithmetic expressions. However, the existing MLN inference techniques cannot effectively support these hybrid rules due to the following two new challenges:

1. Modeling the integration of logic formula and arithmetic expression. We note that the latest approach of Probabilistic Soft Logic (PSL) [10] enables MAP inference on continuous variables over a set of arithmetic rules such as "$r_2 : Performance(p) \leqslant 0.2$" by considering it as a constraint on prior probability. However, it can be observed that an arithmetic expression (e.g., $FastCPU(c) \geqslant 0.9$) is not a predefined continuous logic variable; thus, it cannot be easily integrated into the objective function defined by PSL. Specifically, even though the arithmetic inequalities like "$FastCPU(c) \geqslant 0.9$"

in r_3 can be regarded as a Boolean variable by PSL, computing the truth value of r_3 by the *max* function used in PSL would render its corresponding objective function non-convex. Since the inference of PSL was built on convex optimization, applying PSL inference on r_3 would lead to inaccurate results and convergence failure. Therefore, the existing MLN solutions cannot effectively support the integration of logic formula and arithmetic expression.

2. Scalability. Arithmetic expressions usually involve pair-wise numerical comparison. The existing MLN solutions would generate the combination of all the predicate variables in the grounding process. This results in the undesirable quadratic or even cubic explosion of grounded clauses, which can easily render the inference process unscalable. For instance, consider the rule r_4 in Table 1. The existing inference solutions would result in an n^2 size of clauses for n variables. It is worth pointing out that clause explosion would not only result in inference inefficiency, but also meaningless inference results. In the circumstance of clause explosion, the techniques based on Gibbs sampling [11,12] may fail because the sampler would be trapped in a local state. As shown in our experimental study, the predictions of PSL may become inaccurate because it fails to converge.

Table 1. Examples of knowledge rules.

	Knowledge Rules	Size
r_1	$Frequency(c) \wedge Core(c) \Rightarrow FastCPU(c)$	n
r_2	$Performance(c) \leqslant 0.2$	n
r_3	$FastCPU(c) \geqslant 0.9 \wedge HasCPU(p,c)$ $\wedge Memory(p) \geqslant 0.8 \Rightarrow Performance(p)$	n
r_4	$Performance(p_1) \geqslant Performance(p_2) \wedge Similarprice(p_1,p_2) \Rightarrow$ $Performancecost(p_1) \geqslant Performancecost(p_2)$	n^2

To address the aforementioned challenges, we propose a novel inference framework called the Numerical Markov Logic Network (NMLN). The framework defines the optimization objective of inference as a novel exp-loss function, which can seamlessly integrate logic and arithmetic expressions. We also present an inference approach of exp-loss function decomposition based on convex optimization and use the technique of ADMM (Alternating Direction Method of Multipliers) to parallelize the inference process for improved efficiency. The major contributions of this paper can be summarized as follows:

- We propose a novel probabilistic framework for hybrid knowledge inference. We define the hybrid knowledge rules and present the optimization model.
- We propose a scalable inference approach for the proposed framework based on the decomposition of the exp-loss function.
- We present a parallel solution for hybrid knowledge inference based on convex optimization.
- We empirically evaluate the performance of the proposed framework on real data. Our extensive experiments show that compared to the existing MLN techniques, the proposed approach can achieve better prediction accuracy while significantly reducing inference time.

2. Related Work

Probabilistic Programming Languages (PPLs) [13] seek to separate model specification from inference and learning algorithms, thus making it easy for end users to construct probabilistic models in a simple style. Recent PPL platforms, including PyMC3 [14], Edward [15], and Pyro [16], require that the user defines the model structure such as probabilistic graph models (i.e., represents a joint probability distribution for the problem in hand).

The Markov Logic Network (MLN) [1] was originally proposed for combining first-order logic inference and probabilistic reasoning. Based on the original model, several variants and significant improvements have been proposed. For example, Tuffy [7] was the first system that implemented MLN inference by RDBMS. ProKB [8] proposes a probabilistic knowledge base system allowing uncertain first-order relations and can dramatically reduce the grounding time cost in Tuffy. Deepdive [9] was also an improvement over Tuffy, which has been widely applied to different applications. It provides a powerful knowledge base construction tool and optimizes MLN inference by a combination of statistical inference and machine learning. Our previous work of POOLSIDE [17] proposed a ranking system for commercial products according to their attributes and user comments. Implemented using Deepdive, POOLSIDE provides a naive predefined function to specify the relations between attribute values. The recently proposed variant Quantified Markov Logic Networks (QMLNs) [18] extends classical MLN with a statical quantifiers $\forall^*$, which provides a kind of quantification describing for example most, few, or at least k thresholds. More recently, Flash [19] exploited MLN to express the Spatial Probabilistic Graphical Model (SPGM), which can perform SPGM predictions efficiently. The MLN has been widely applied to various areas, including activity recognition systems in smart homes [20], root cause analysis in IT infrastructure [21], and natural language understanding [22], to name a few. Note that these systems were all designed for inference on first-order logic rules, but they cannot effectively support the inference on hybrid knowledge rules.

The latest research is mainly focused on the applications. MLNClean [23] was proposed for data cleaning, which is able to clean both schema-level and instance-level errors. The authors of SMLN [24] proposed a framework with native support for spatial data. The paper [25] proposed R-KG, a robot intelligent service, to reason about knowledge based on a Markov logic network.

On the issue of probabilistic reasoning, the MLN mainly focuses on two aspects: inference optimization and model learning. The traditional MLN-based inference techniques suffer from the issue of scalability due to their dependence on the generative model, which embeds all the data and targets in a model. The lifted inference technique [26] was proposed to simplify the MLN network by exploiting symmetry in the model. The authors of [27] proposed a technique to enable large-scale parallel inference by making Gibbs sampling work on the divided networks. The authors of [28] also proposed a query-driven technique that can leverage the local network for query prediction. Moreover, in our previous work POOLSIDE [17], we also proposed an improved query-driven inference algorithm, which exploits the information in the known neighbors to predict the query node. Ground Network Sampling (GNS) [29] proposed in 2016 offers a new instantiation perspective, which can ground from a set of sampled paths at inference time; thus, GNS offers better scalability compared to MLN. Model learning for the MLN includes parameter learning and structural learning. Parameter learning aims to find the optimal weights for a set of rules. This is usually achieved by optimizing different metrics of the objective function [30–32]. Structure learning instead aims to learn both logic formulas and their weights, which use the top-down [33] or bottom-up [34] search strategy to find formulas. The authors of [35] proposed a functional-gradient boosting algorithm that learns parameter and structure simultaneously. Since feature representation using neural networks has received much attention from researchers in various domains, neural Markov logic networks [36] also propose to learn the implicit representation of rules using neural networks instead of the explicit rules specified by humans.

To represent fuzzy logic, the MLN models have been extended from the binary field to the continuous field. The hybrid MLN [37] defines and reasons about the soft equality and inequality constraints for first-order relations. Probabilistic Soft Logic (PSL) [10] extends binary variables in the MLN into the continuous range $[0, 1]$. PSL uses Lukasiewicz logic [38] to compute the truth values of logic clauses. Moreover, PSL allows users to define arithmetic rules, which can be interpreted as constraints on the variables, and transforms

the MAP inference into a convex optimization problem. With the help of ADMM [39], the inference can be effectively parallelized and scaled up well to the data size.

However, PSL cannot effectively support the inference on hybrid knowledge rules; the proposed inference techniques thus cannot address the clause explosion issue.

3. Hybrid Knowledge Rules

The first-order relation consists of a predicate and several predicate variables, e.g., "$relation(y_1, y_2)$", where the "relation" is called a predicate, which represents the relationship between variables, while y_1 and y_2 are called predicate variables. If we replace the predicate variables of a relation with the instance data, the relation can be considered as grounded. In our inference system, each grounded relation is regarded as an inference variable or evidence, which has a truth value at $[0, 1]$ intervals, to indicate whether the relation is held (equal to one) or not (equal to zero).

A hybrid knowledge rule involves both arithmetic and logic expressions. Formally, we define a hybrid knowledge rule by extending the definition of the knowledge rule [10] as follows:

Definition 1. *Suppose that* $\mathbf{x}$ *denotes the set of first-order relation variables and* $\ell(\mathbf{x})$ *denotes a linear function, which consists of variables in* $\mathbf{x}$*. A hybrid knowledge rule, r, can be represented by a disjunction form of:*

$$t_1 \vee t_2 \vee \cdots \vee t_n, \tag{1}$$

where t_i *denotes a term, which should be one of the following three types:*

- *(1)* t_i *is a first-order relation* x *or its negation* $\neg x$*, where* $x \in \mathbf{x}$*;*
- *(2)* t_i *is a logic expression, and* $\mathbf{x_i}$ *denotes its variables, where* $\mathbf{x_i} \subseteq \mathbf{x}$*;*
- *(3)* t_i *is a linear inequality in the form of* $\ell(\mathbf{x_i}) \leqslant (or \geqslant) 0$*, where* $\mathbf{x_i} \subseteq \mathbf{x}$*.*

4. Inference Framework

To introduce our inference framework, we first define the knowledge inference problem as follows:

Definition 2. *Suppose that* $\mathbf{r}$ *denotes the set of knowledge rules,* $\mathbf{x}$ *denotes a set of variables (including the set of inference variables* V *and the set of evidence* Λ*), and* Φ_j *denotes a function defined over variables* $\mathbf{x}$*, which represents the constraint based on the rule* $r_j \in \mathbf{r}$*. The knowledge inference problem is to find a solution* $V^\star$ *for the variables, such that:*

$$V^\star = \underset{\mathbf{V} \in [0,1]^n}{argmin} \sum_{r_j \in \mathbf{r}} \Phi_j(\mathbf{x}) \tag{2}$$

In order to define Φ_j, we use Lukasiewicz logic [38], which extends binary variables to the continuous field $[0, 1]$, to represent the logic formula. Lukasiewicz logic transforms a logic operator in the following manner:

$$x_1 \wedge x_2 \Rightarrow max(x_1 + x_2 - 1, 0) \tag{3}$$

$$x_1 \vee x_2 \Rightarrow min(x_1 + x_2, 1) \tag{4}$$

$$\neg x \Rightarrow 1 - x \tag{5}$$

Note that the latest approach of PSL can handle the clauses containing only logic formulas. Based on Lukasiewicz logic, PSL transforms a logic formula into a linear inequality $\ell(\mathbf{x}) \leqslant 0$, where:

$$\ell(\mathbf{x}) = \sum_{x_i \in \mathbf{x}} \beta_i x_i + b \tag{6}$$

is a linear function, which defines the distance of a constraint from being satisfied. Given a logic formula (rule) r in disjunctive form, let $I^- \subseteq \mathbf{x}$ and $I^+ \subseteq \mathbf{x}$ denote the set of variables

with and without the negation prefix "¬", respectively. Formally, the linear function $\ell(\mathbf{x})$ can be represented by:

$$\ell(\mathbf{x}) = 1 - \sum_{x_i \in I^+} x_i - \sum_{x_i \in I^-} (1 - x_i). \tag{7}$$

Based on the transformation, it then defines a Hinge-Loss Markov Random Field (HF-MRF), which extends the MLN to the continuous field. The loss function for each clause can be formally represented by:

$$\phi(\mathbf{x}) = (max\{\ell(\mathbf{x}), 0\})^p, \tag{8}$$

where $\mathbf{x}$ denotes the vector of variables, p denotes a user-defined parameter, and $\ell(\mathbf{x})$ denotes the linear function, as shown in Equation (7).

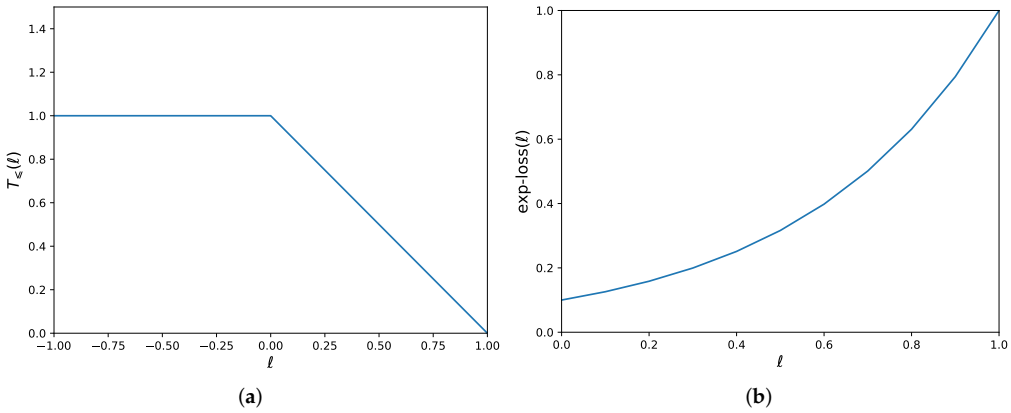
Unfortunately, the loss function as defined in Equation (8) cannot handle a hybrid knowledge rule involving both a logic formula and arithmetic inequalities. It can be observed that directly modeling the inference of hybrid knowledge rules by Equation (8) would render its corresponding objective function non-convex.

To integrate all terms in a hybrid rule into a function, we consider the truth value of each arithmetic expression (inequalities) as a continuous logic variable in the interval of $[0, 1]$, which is consistent with its semantic and logic propositions. Formally, we define the truth value for a linear inequality, $\ell \leqslant 0$, as follows:

$$T_{\leqslant}(\ell) = min\left(1 - \frac{\ell}{sup(\ell)}, 1\right), \tag{9}$$

$$sup(\ell) = \sum_{x_i \in I^+} \beta_{x_i} + c, \tag{10}$$

where $sup(\ell)$ denotes the sum of all positive variables' coefficients β_{x_i} and constant c. Note that the linear inequality of $\ell \geqslant 0$ can be equivalently transformed into $-\ell \leqslant 0$. Figure 1a demonstrates the functional relation between a linear function value ℓ and its truth value. As shown in the figure, with a linear inequality being normalized by its supremum, its truth value is equal to the maximal value of one when the inequality is satisfied, and it decreases to zero as the violation reaches the maximum level.



(**a**) (**b**)

Figure 1. Metric functions: an example: (**a**) truth value function; (**b**) exp-loss function.

It is noteworthy that the truth value as defined in Equation (9) is consistent with the PSL transformation with regard to Equations (3) and (4). For the negation operator, we define:

$$T(\neg(\ell \leqslant 0)) = T(-\ell \leqslant 0), \tag{11}$$

Our inference framework then defines a linear function ℓ for a hybrid rule as follows:

$$\ell(\mathbf{x}) = 1 - \sum_{x_i \in I^+} x_i - \sum_{x_i \in I^-} (1 - x_i) - \sum_{\ell_i \in \ell^{\leqslant}} \left(1 - \frac{\ell_i}{sup(\ell_i)}\right), \tag{12}$$

where $\ell^{\leqslant}$ denotes the set of linear inequalities in the rule.

Note that the hybrid rules can be directly converted to the PSL loss function formulated in Equation (8), by replacing the linear functions in Equation (7) with Equation (12), such that hybrid rules' inference can be solved by PSL, as we did in our empirical evaluation study. However, such an inference approach causes the clause explosion problem, as we discussed in the introduction. To solve the problem of clause explosion, we instead define an exp-loss function to measure the violation of a rule as follows.

Definition 3. *Let $\ell(\mathbf{x})$ denote the linear function defined in Equation (12), and $\alpha > 1$ denote the base argument, which can be e or other constants. The exp-loss function is defined by:*

$$\alpha^{\ell(\mathbf{x})-1}. \tag{13}$$

Lemma 1. *$\alpha^{\ell(\mathbf{x})-1}$ is convex when $\alpha > 1$.*

Proof. A function twice differentiable is convex iff the Hessian matrix is positive semi-definite. Take:

$$f(x) = \alpha^{l(x)-1} = \alpha^{\sum_{i=1}^{n} \beta_i x_i + d - 1}.$$

Computing the Hessian:

$$\frac{\partial^2 f(x)}{\partial x_i \partial x_j} = \beta_i \beta_j \alpha^{\sum_{i=1}^{n} \beta_i x_i + d - 1},$$

we see it is actually positive semi-definite, because for any λ_i,

$$\sum_{i,j} \beta_i \beta_j \alpha^{\sum_{i=1}^{n} \beta_i x_i + d - 1} \lambda_i \lambda_j = (\sum_{i} \lambda_i \alpha^{\sum_{i=1}^{n} \beta_i x_i + d - 1})^2 \geqslant 0. \tag{14}$$

□

It is worth pointing out that we chose the exp-loss function to measure the violation of a rule due to following reasons:

1. The exp-loss is a natural extension to the hinge-loss function defined in Equation (8). The exponential power $\ell(\mathbf{x})$ guarantees a greater loss when a violation of the rule occurs. On the other hand, it can be observed that even though the function is not zero when the rule is satisfied (e.g., if $\alpha = e$, the loss is e^{-1} if the rule is satisfied), the value of the exp-loss and its gradient becomes very small in the negative interval, which can be considered as a soft constraint of the $max()$ function.
2. As shown in the following section, the exp-loss function enables the scalable inference based on function decomposition. It can effectively address the challenge of the explosion of grounded clauses.

Let V denote the set of unknown variables for inference. Given a set of hybrid knowledge rules $\mathbf{r}$ and the weight w_j with respect to $r_j \in \mathbf{r}$, the inference target is to minimize the sum of all weighted loss functions generated by all clauses as follows:

$$\underset{\mathbf{V} \in [0,1]^n}{argmin} \sum_{j=1}^{|\mathbf{r}|} w_j \sum_{i=1}^{|\mathfrak{g}(r_j)|} \alpha^{\ell_j(\mathbf{x}_i)-1}. \tag{15}$$

where $\mathfrak{g}()$ denotes the operation of grounding and $\mathbf{x}_i$ denotes the set of variables in the i-th clause. According to Equation (15), each rule r_j has the size of $|\mathfrak{g}(r_j)|$ clauses in its loss, which are generated by replacing the predicate variables in first-order relations with the possible instances in the data. This process is known as grounding in the existing

MLN solution, which is usually implemented by a series of database join operations. Our framework performs grounding for inference optimization, while the MLN performs grounding to generate a factor graph.

5. Inference Optimization

5.1. Decomposition of Exponential Loss Function

In the scenario of hybrid knowledge inference, grounding the rules, which involves numerical value comparison between two predicate variables, such as the term "$Performance(p_1) \leqslant Performance(p_2)$", could easily result in clause explosion. To address this issue, our solution first decomposes the rule relations into groups and then grounds them separately. We illustrate the replacement process by a simple example as follows:

Example 1. *Given the rule of* $Frequency(y_1) \geqslant Frequency(y_2) \Rightarrow FastCPU(y_1) \geqslant FastCPU(y_2)$, *its loss function (according to Equations* (12) *and* (13)*) can be represented by:*

$$\alpha^{Fr(y_2)-Fr(y_1)+Fc(y_1)-Fc(y_2)-2}, \tag{16}$$

where Fr and Fc denote the predicates frequency and FastCPU, respectively. The total loss of the rule is estimated by the sum of all the grounded loss functions as follows:

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha^{Fr(y_j)-Fr(y_i)+Fc(y_i)-Fc(y_j)-2}. \tag{17}$$

It is noteworthy that the total sum of the loss can be decomposed into:

$$\left(\sum_{i=1}^{n}\alpha^{-Fr(y_i)+Fc(y_i)}\right)\left(\sum_{j=1}^{n}\alpha^{Fr(y_j)-Fc(y_j)}\right)\alpha^{-2}. \tag{18}$$

Suppose that y_i *has n instances. Compared to the original form in Equation* (16)*, which requires a computational time of* $O(n^2)$*, computing the loss function in Equation* (18) *only requires* $O(2n)$.

In the general case, where the hybrid rules may contain facts and share common variables, the decomposition may be more complicated. Formally, we define the irreducible groups as follows.

Definition 4. *Suppose that a rule r contains the relations* $\mathbf{R} = \{R_1, \cdots, R_m\}$ *and* $\mathbf{y_i} = (y_1^i, \cdots, y_k^i)$ *denotes the variables in* R_i. *We call* R_i *irreducible if* $\forall R_j \neq R_i$, $\mathbf{y_i} \nsubseteq \mathbf{y_j}$; *otherwise, there exists a relation* R_j *with* $\mathbf{y_i} \subseteq \mathbf{y_j}$, *and* R_i *can be reduced to* R_j. *An irreducible group consists of an irreducible relation* R_i *and all the relations reducible to* R_i, *and we denote it by* $\hat{R}_i$. *The set of predicate variables shared by two or more irreducible relations is called a joint variable set, denoted by S.*

For the decomposition of the exp-loss function, we first split a hybrid rule into multiple irreducible groups. We sketch the procedure for identifying all the irreducible groups and their joint variables in Algorithm 1. For each relation R_i, we can find its irreducible group $\hat{R}_i$ if the relation exists in the groups. Note that a relation might be reduced to more than one irreducible group. However, it can only be assigned to one group. The algorithm simply assigns it to the first irreducible group it meets. An illustrative example of how to split a set of relations into irreducible groups is also shown in Figure 2. In the example, the relations $R(y_1)$ and $R(y_3)$ can be reduced to the relations $R(y_1, y_2)$ and $R(y_2, y_3)$, respectively. The splitting operation results in totally three irreducible groups. It can be observed that the relations $R(y_1, y_2)$ and $R(y_2, y_3)$ share the variable y_2, and $R(y_4)$ is disjoint to both $R(y_1, y_2)$ and $R(y_2, y_3)$.

Algorithm 1: Find irreducible groups and joint variables.

Input: relations set $\mathbf{R} = \{R_1, \cdots, R_m\}$ and predicate variable set $\mathbf{y_i} = \left(y_1^i, \cdots, y_k^i\right)$ with respect to R_i
Output: irreducible groups $\hat{\mathbf{R}}$ and joint variable set $\mathbf{S}$.
$\hat{\mathbf{R}} = \{\{R_1\}, \cdots, \{R_m\}\}$ and $\mathbf{S} = \emptyset$;
for R_i *in* $\mathbf{R}$ **do**
  for R_j *in* $\mathbf{R} - R_i$ **do**
    if $\mathbf{y}_i \subseteq \mathbf{y}_j$ **then**
      find $\hat{R}_{i'} \in \hat{\mathbf{R}}$ where $R_i \in \hat{R}_{i'}$;
      find $\hat{R}_{j'} \in \hat{\mathbf{R}}$ where $R_j \in \hat{R}_{j'}$;
      merge the set $\hat{R}_{i'}$ and $\hat{R}_{j'}$;
    end
  end
end
for $\hat{R}_i$ *in* $\hat{\mathbf{R}}$ **do**
  for $\hat{R}_j$ *in* $\hat{\mathbf{R}} - \hat{R}_i$ **do**
    if $S_{ij} = \mathbf{y_i} \cap \mathbf{y_j} \neq \varnothing$ **then**
      add S_{ij} to $\mathbf{S}$;
    end
  end
end

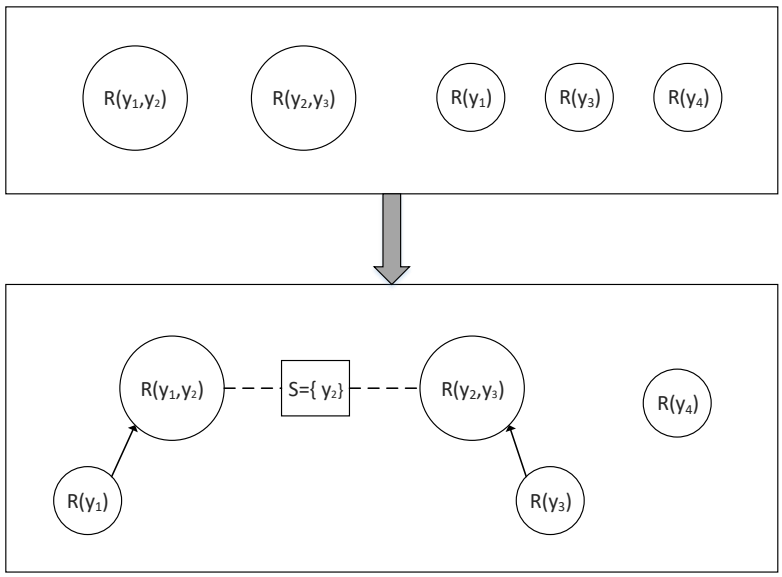


Figure 2. Example of relation decomposition.

Now, we are ready to describe how to leverage irreducible groups $\hat{\mathbf{R}}$ for decomposition optimization. In the proposed inference framework, the first-order relation is represented by a linear function in the exponential term in a loss function. Suppose that $\hat{\mathbf{R}}$ has k irreducible groups, which is denoted by $\hat{R}_j$. Then, the linear function $\ell(\mathbf{x})$ can be split into $k+1$ parts $\{\ell_1, \cdots, \ell_k, \ell_c\}$, where ℓ_j is the variables and their coefficients corresponding to the relations in $\hat{R}_j$, and ℓ_c is the constant part. Therefore, the loss function can be reformulated as follows:

$$loss(r) = w \sum_{i=1}^{|\mathfrak{g}(r)|} \alpha^{\ell(\mathbf{x}_i)-1} = w \sum_{i=1}^{|\mathfrak{g}(r)|} \prod_{j=1}^{k} \alpha^{\ell_j(\mathbf{x}_{ij})} \cdot \alpha^c, \tag{19}$$

where $\mathbf{x}_{ij}$ denotes the variables respecting the i-th grounded relations in $\hat{R}_j$. To decompose the loss function, we first split all the clauses $\mathfrak{g}(r)$ that share the same grounded relation in the set of joint variables S into partitions. In each partition, the grounding clause is the combination of all variables in the irreducible groups. As a result, the sum of clauses in a partition can be represented by the product of all the sums in each group. Without loss of generality, we assume all irreducible relations have n instances, and the set of joint variables S has θ instances. The decomposed loss function can be stated as follows:

$$loss(r) = w \sum_{s=1}^{\theta} \prod_{j=1}^{k} \left(\sum_{i=1}^{n} \alpha^{\ell_j(\mathbf{x}_{sij})} \right) \cdot \alpha^{c}. \tag{20}$$

Now, we estimate the complexity of loss computation. The original loss computes all combinations of clauses of irreducible relations, which is $O(\theta n^k)$. As shown in Equation (20), our proposed technique of function decomposition can reduce the computational complexity from $O(\theta n^k)$ to $O(\theta nk)$.

It is noteworthy that the grouped loss function is just a deformation of the original loss function. Each rule in the form of Equation (19) can be converted to Equation (20). According to Equations (12), (13) and (15), the expansion of the loss function is the sum of exponential functions, and all exponential functions have a linear exponent. As a result, the loss function is convex. Our proposed method can effectively find the global optimal.

Now, we provide the entire process of hybrid knowledge rule inference in Algorithm 2. The algorithm first generates variables that represent the first-order relations in the dataset and then grounds the clauses for each rule r_j in the form of decomposed exp-loss functions. Finally, we use the ADMM algorithm introduced in the following subsection to optimize the sum of losses for all knowledge rules.

Algorithm 2: Inference of hybrid knowledge rules.

Input: set of hybrid knowledge rules $\mathbf{r}$, relation set $\mathbf{R} = \{R_1, \cdots, R_m\}$, predicate variable set $\mathbf{y_i} = (y_1^i, \cdots, y_k^i)$ with respect to R_i, and the instances of dataset D.

Output: Solution $V^\star \in [0,1]^n$ for the inference variables V.

Generate the set of variables $\mathbf{x}$ according to $\mathbf{R}$ and D;

for $r_j \in \mathbf{r}$ **do**

Find irreducible groups and joint variables for r_j by Algorithm 1;

Generate the $loss(r_j)$ for r_j (grounding) in the form of Equation (20);

end

Find the optimal solution $V^\star = \underset{\mathbf{V} \in [0,1]^n}{argmin} \sum_{r_j \in \mathbf{r}} loss(r_j)$

We provide an example of the comparison between our framework and PSL in the Figure 3. In this example, we selected hybrid knowledge used in our experimental study for entity linking, to demonstrate the loss functions in three scenarios: the original PSL hinge-loss and the exp-loss with and without loss decomposition. As shown in the figure, the rule consists of two first-order relations. Each relation in the dataset has three instances. Since PSL does not support hybrid rule inference, we show the loss function when the linear inequality is directly regarded as a logical variable. It is easy to observe that the original PSL loss function cannot guarantee convexity. The decomposed exp-loss function reduces the number of clauses from 3^2 to 2×3.

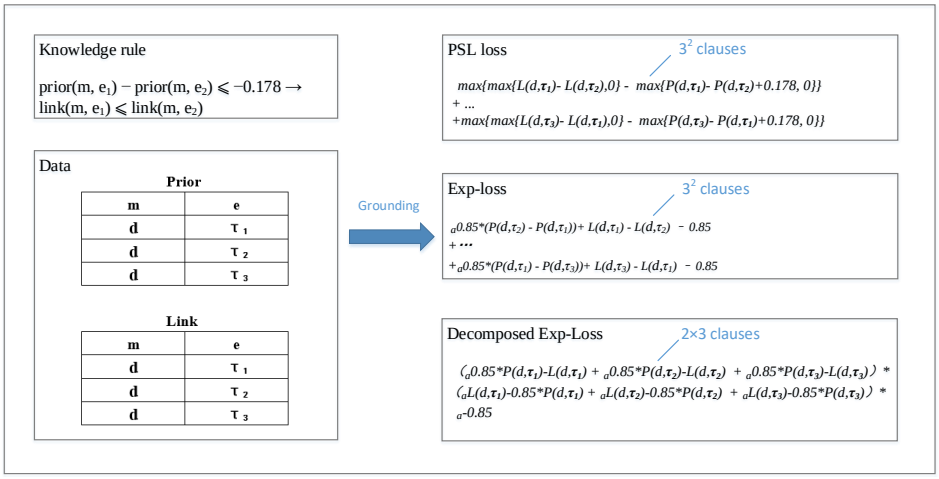


Figure 3. Example of the comparison of the proposed framework. PSL, Probabilistic Soft Logic.

5.2. Parallel Optimization

Our decomposition-based method can effectively compute the loss function proposed in Equation (19). In this subsection, we demonstrate how to implement our method in the optimization process. In order to achieve efficient inference, we use the approach of parallel optimization based on the ADMM algorithm. ADMM is a distributed optimization technique that focuses on solving large-scale convex optimization problems. It is generally applicable to the loss function in the form of $\sum_{i=1}^{N} f_i(x)$, where each term $f_i(x)$ is a convex function. The main idea of ADMM is to replace the variables in each term with independent local variables and add the constraints on these variables by the augmented Lagrange method. ADMM iteratively optimizes the local variables and updates the consensus global variables until they converge. More details about ADMM optimization is shown in [39].

The total loss function of the inference is the sum of all clauses in the form of decomposed exp-loss functions. For simplicity, we define:

$$P(\mathbf{x}) = w \prod_{j=1}^{k} \left(\sum_{i=1}^{n} \alpha^{\ell_j(\mathbf{x}_{ij})} \right) \cdot \alpha^{c} \tag{21}$$

as a term of the loss function, such that the total loss is the sum of all terms, which can be formulated as follows:

$$loss(\mathbf{x}) = \sum_{h=1}^{H} P_h(\mathbf{x}_h). \tag{22}$$

where H is the size of all terms in the loss function. By reformulating the optimization problem with local variables and related constraints by the augmented Lagrange function, ADMM transforms the MAP problem into:

$$loss(\mathbf{z}, \gamma, \mathbf{x}) = \sum_{h=1}^{H} P_h(\mathbf{z}_h) + \sum_{h=1}^{H} \gamma_h^{\top} (\mathbf{z}_h - \mathbf{x}_h) + \sum_{h=1}^{H} \frac{\rho}{2} \|\mathbf{z}_h - \mathbf{x}_h\|_2^2, \tag{23}$$

where $\mathbf{z}_h$ denotes a copy of the variables in $\mathbf{x}_h$, $\mathbf{x}_h$ denotes the variables in $\mathbf{x}$ that correspond to $\mathbf{z}_h$, γ denotes the vector of Lagrange multipliers, and $\rho > 0$ denotes the step-size parameter. Each set of local variables in $\mathbf{z}$ is independent of the others, such that for any two sets of local variables $\mathbf{z}_h$ and $\mathbf{z}_{h'}$, $\mathbf{z}_h \cap \mathbf{z}_{h'} = \varnothing$.

The optimization process iteratively updates the following three blocks until it converges:

$$\gamma_h^t \leftarrow \gamma_h^{t-1} + \rho \left(\mathbf{z}_h^{t-1} - \mathbf{x}_h^{t-1} \right), \forall h = 1, \ldots, H \tag{24}$$

$$\mathbf{z}^t \leftarrow \underset{\mathbf{z}}{argmin}\ \ loss\Big(\mathbf{z}, \gamma^t, \mathbf{x}^{t-1}\Big), \tag{25}$$

$$\mathbf{x}^t \leftarrow \underset{\mathbf{x}}{argmin}\ \ loss\big(\mathbf{z}^t, \gamma^t, \mathbf{x}\big). \tag{26}$$

The optimization process converges if the local variables converge to the global variables and the global variables converge at the last iteration. Specifically, the two convergence conditions can be represented by:

$$\left\|\bar{\mathbf{r}}^t\right\|_2 = \left(\sum_{h=1}^{H}\left\|\mathbf{z}_h^t - \mathbf{x}_h^t\right\|_2^2\right)^{\frac{1}{2}} \leqslant \epsilon^{pri}, \tag{27}$$

and:

$$\left\|\bar{\mathbf{s}}^t\right\|_2 = \rho\left(\sum_{i=1}^{m} K_i\left\|x_i^t - x_i^{t-1}\right\|_2^2\right)^{\frac{1}{2}} \leqslant \epsilon^{dual}, \tag{28}$$

where m denotes the total number of variables in $\mathbf{x}$ and $\left\|\bar{\mathbf{r}}^t\right\|_2$ and $\left\|\bar{\mathbf{s}}^t\right\|_2$ denote primal residual and dual residual, respectively. ϵ^{pri} and ϵ^{dual} are feasibility tolerances for the primal and dual feasibility conditions, and K_i is the number of local variables for a variable x_i.

Our optimization takes the same steps as shown in Equations (24) and (26). For Equation (25), we follow the traditional ADMM practice to apply the parallel optimization to each clause. However, for each clause, our method does not find the minimal result at each iteration. Instead, it iteratively updates each local variable to its minimal value while fixing the values for other variables. The gradient $\bigtriangledown loss(\mathbf{z})$ corresponds to the vector composed by the first derivative of each element. Since the local variables are independent for each term:

$$P(\mathbf{z}, \gamma, \mathbf{x}) = w\prod_{j=1}^{k}\left(\sum_{i=1}^{n}\alpha^{\ell_j(\mathbf{z}_{ij})}\right)\cdot\alpha^c + \gamma^\top(\mathbf{z}-\mathbf{x}) + \frac{\rho}{2}\|\mathbf{z}-\mathbf{x}\|_2^2, \tag{29}$$

such that we only demonstrate the gradient for a single term. Let $z_{ij'}$ denote a local variable, which belongs to the irreducible group $\hat{R}_{j'}$. The gradient of $z_{ij'}$ can be represented by:

$$\frac{\partial P(\mathbf{z}, \gamma, \mathbf{x})}{\partial z_{ij'}} = \ln\alpha\cdot\beta\cdot\alpha^{\ell_{j'}(\mathbf{z}_{ij'})}w\prod_{j=1}^{k}\left(\sum_{i=1}^{n}\alpha^{\ell_j(\mathbf{z}_{ij})}\right)\cdot\alpha^c + \gamma_{ij'} + \rho\Big(z_{ij'} - x_{ij'}\Big), \quad j \neq j' \tag{30}$$

where $\mathbf{z}_{ij'}$ denotes the set of variables that are in the same group with $z_{ij'}$. For the computation of the first term $\prod_{j=1}^{k}\left(\sum_{i=1}^{n}\alpha^{\ell_j(\mathbf{z}_{ij})}\right)\cdot\alpha^c$, it is obvious that each $z_{ij'}$ in the same group $R_{j'}$ shares the same product from $k-1$ groups. Let f_j denote $\sum_{i=1}^{n}\alpha^{\ell_j(\mathbf{z}_{ij})}$, such that:

$$P(\mathbf{z}) = w\prod_{j=1}^{k} f_j\cdot\alpha^c = w\prod_{j=1}^{k}\left(\sum_{i=1}^{n}\alpha^{\ell_j(\mathbf{z}_{ij})}\right)\cdot\alpha^c. \tag{31}$$

In order to compute the gradient, we first compute the product $P(\mathbf{z})$ and then compute the gradient for each variable as follows:

$$\frac{\partial loss}{\partial z_{ij'}} = \ln\alpha\cdot\beta\cdot\alpha^{\ell_{j'}(\mathbf{z}_{ij'})}\cdot P(\mathbf{z})\cdot\frac{1}{f_{j'}} + \rho\Big(z_{ij'} - x_{ij'}\Big), \quad j \neq j'. \tag{32}$$

Equation (32) significantly reduces the computation in the optimization process, by sharing the product $P(\mathbf{z})$ for every variable.

6. Experimental Study

In this section, we empirically evaluate the performance of the proposed solution by a comparative study. We compare the NMLN to PSL, which is the state-of-the-art technique for soft logic inference. PSL has been empirically shown to have the best performance on MLN inference among the existing solutions. More importantly, to the best of our knowledge, it is the only technique that is able to infer hybrid knowledge rules, even though it cannot solve the issue of clause explosion. To enable PSL inferences on hybrid knowledge rules, we replace the linear functions in Equation (7) by Equation (12), such that the rule can be converted to a linear function, which can then be solved by PSL inference. It is noteworthy that other Gibbs sampling-based methods such as Deepdive fail on hybrid rules due to the existence of extremely high-probability states. The sampler would be trapped in a local state, which requires an unacceptable time to sample the correct distribution. We evaluated the performance of different techniques on two real applications: mobile phone ranking and entity linking. We show the statistics of the datasets in Table 2.

Table 2. Statistics of the test datasets.

Dataset	Total No. of Variables	No. of Non-Matches	No. of Matches
Mobile Phone	1058	–	–
AIDA-CONLL	728,225	713,113	15,112
Wiki-Sport	28,244	24,244	4000
Wiki-FourDomains	23,828	19,318	4510

6.1. Comparative Study

In the comparative study, we set the number of parallel threads at 6, $\epsilon^{pri} = 10^{-3}$, and $\epsilon^{dual} = 10^{-5}$ in all experiments. For the NMLN, we set the base of exponential function $\alpha = e$ and step size $\rho = 0.5$ as the default.

Mobile phone ranking: For this experiment, we needed to rank various mobile phones by performance for users. Since the performance evaluation of mobile phones is to some extent a subjective problem, it is difficult to obtain the ground truth. Therefore, we extracted the phone's ranking list from a well-known benchmark website. Available online: (https://benchmarks.ul.com/ accessed on 3 June 2018), which also lists the specific details of phones such as the CPU, memory, or size. We considered the positions of phones in the ranking list as the annotations to evaluate the inference results. The test dataset contained 899 smart phones. We define the average distance to evaluate the quality of the inference results:

$$D(\mathbf{r}, \mathbf{r}^*) = 1 - \frac{1}{N^2} \sum_{i=1}^{N} |r_i - r_i^*|, \tag{33}$$

where $\mathbf{r}$ denotes the results ranked by inference and $\mathbf{r}^*$ denotes the annotations in the ranking list. This function takes the maximal value of one when the inference results are exactly the same as the annotations. We defined six rules, which were presented in Appendix A, for performance inference. The detailed results are presented in Table 3. They evidently show that the NMLN achieves similar performance to PSL on prediction accuracy, while it requires significantly less inference time. The two methods have a similar accuracy due the rules used in this task being simple; thus, PSL can also give a fine prediction.

Table 3. Evaluation results on mobile phone ranking. NMLN, Numerical Markov Logic Network.

	Distance_avg	Grounding	Inference	Total
NMLN	0.857	0.13 s	0.45 s	2.09 s
PSL	0.853	34.7 s	37.9 s	73.9 s

Entity linking: Our empirical study was conducted on three real benchmark datasets, whose details are described as follows.

- AIDA-CONLL: This dataset was constructed based on the source of CONLL2003 [40], which contains 1393 news articles. It consists of proper noun annotations, which indicate its corresponding entities in YAGO2 [41]. In our experiments, we evaluated all approaches on its testB dataset.
- Wiki-Sports: This dataset contains the articles on the topic of sports extracted under the feature article page in Wikipedia. The mentions in the dataset are extracted from the anchor texts in the articles and annotated by the entities to which they link. We used the disambiguation page of Wikipedia to generate the candidates for each mention. In order to avoid the leakage of label information, we eliminated the corresponding Wiki pages during the extracting link text for the entities.
- Wiki-FourDomains: This dataset contains the articles extracted on four topics, which include films, music, novels, and television episodes, on Wikipedia. We applied the same process on the dataset as Wiki-Sports to generate mention-annotations and candidate entities.

In the experiment, we linked a mention in the articles to the YAGO2 entity with the highest inference probability. We first extracted the following six features from the YAGO2 knowledge-base: prior, semantic similarity, coherence, syntax similarity, edit distance, and word2vector similarity. Note that we also eliminated the mention-entity pair candidates, which are obviously not matched, from the inference process. Otherwise, the large number of candidates may cause PSL memory to overflow.

To show the inference capability on a set of decision rules, we made use of the annotations from 300 documents to train a random forest. For each leaf node in the forest, we generated a decision rule, which was formulated as the logic implication of "$X \rightarrow Y$", where Y is the leaf node and X is the logic conjunction of all decision nodes in the path from the root to Y. We retained in total 38 rules whose impurity (measured by Gini) was less than 0.025. In addition, we added the rule of $link(m, e) \leqslant 0.2$ for every target pair such that the candidates unconstrained by any rule can take a small value. The rules were presented in Appendix B.

The detailed evaluation results are presented in Table 4. It can be observed that the NMLN performs considerably better than PSL on prediction accuracy. The experiment showed that PSL cannot converge to consensus values; thus, it cannot perform well. On inference efficiency, the NMLN also performed considerably better than PSL: the NMLN ends within half an hour, while PSL takes more than 14 h.

Table 4. Evaluation results on entity linking.

	In-KBacc	**Grounding**	**Inference**	**Total**
AIDA-CNOLL				
NMLN	0.805	278 s	1344 s	1745 s
PSL	0.708	25,636 s	25,566 s	51,661 s
Wiki-Sport				
NMLN	0.865	23 s	162s	201 s
PSL	0.826	1889 s	889 s	2793 s
Wiki-FourDomains				
NMLN	0.893	14 s	138 s	164 s
PSL	0.876	1196 s	545 s	1753 s

Now, we provide an analysis of the experimental results. As mentioned in Equation (29), for each term $P(\mathbf{x})$ in the loss function, ADMM transforms the term into $P(\mathbf{z}, \gamma, \mathbf{x})$, by replacing $\mathbf{x}$ with local variables z and adds constraints to ensure the local variables converge

to $\mathbf{x}$. Assume that $P(\mathbf{x})$ contains n variables and k irreducible groups. The size of the local variables in $P(\mathbf{z}, \gamma, \mathbf{x})$ is $n \times k$. However, the original form in PSL makes the ADMM method construct n^k local variables, which means that each global variable x_i has k copies in the NMLN, but n^{k-1} copies in PSL. As a result, although the solution found by PSL is the global optimal for the dual problem in ADMM, its local variables actually do not converge to $\mathbf{x}$; thus, the NMLN outperformed PSL on all datasets.

Evaluation of convergence:

In this experiment, we compared the convergence of the two methods on the task of mobile phone ranking. The evaluation results are presented in Figure 4. According to Equations (27) and (28), the optimization process converges if primal residual $\left\|\bar{\mathbf{r}}^t\right\|_2$ and dual residual $\left\|\bar{\mathbf{s}}^t\right\|_2$ are approximately close to zero. It can be observed that the NMLN is able to converge quickly and stably for both conditions. In Figure 4a, the primal residual of PSL stops decreasing at the value of 64, such that the method cannot converge for both conditions.

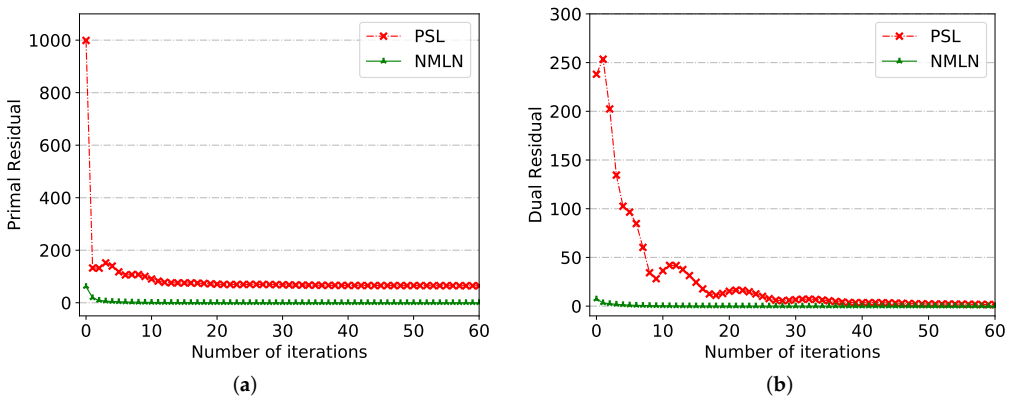


Figure 4. Evaluation of convergence on NMLN vs. PSL: (**a**) primal residual; (**b**) dual residual.

6.2. Scalability

To evaluate the scalability for the NMLN, we generated synthetic data with various sizes for phone ranking inference. The detailed evaluation results are presented in Figure 5. Since the rules contain two kinds of unknown variables $FastCPU(c)$ and $Performance(p)$, we generated the relations at a ratio of 0.2:0.8. Our experiments show that PSL consumes a large amount of memory. The performance of PSL falls dramatically due to memory overflow when the size of the variable exceeds 4500. Compared to PSL, the NMLN scales much better when the data size increases. As shown in Figure 5a, all inference tasks are finished within two seconds by the NMLN. The NMLN spends most of the time in pre-processing when it runs on small data, such that the runtime does not increase significantly in (a). When the data size is large (more than 5000), we also provide the log scale performance in Figure 5b. It can be observed that the runtime scales in an approximately linear fashion. In the figure, the speed has a slight slow down when the data size is greater than 10,000, which is caused by the sequential operations in the pre-processing phase.

We also present the number of iterations required by both techniques to converge in Figure 5c,d. It can be observed that the NMLN takes 36 iterations on all the tasks with the number of variables varying from 100 to 10 M. The reason is that the average size of the local variables with respect to the same global variables is always a fixed number in the NMLN. In PSL, clause explosion causes a single variable taking more local copies when the size increases.

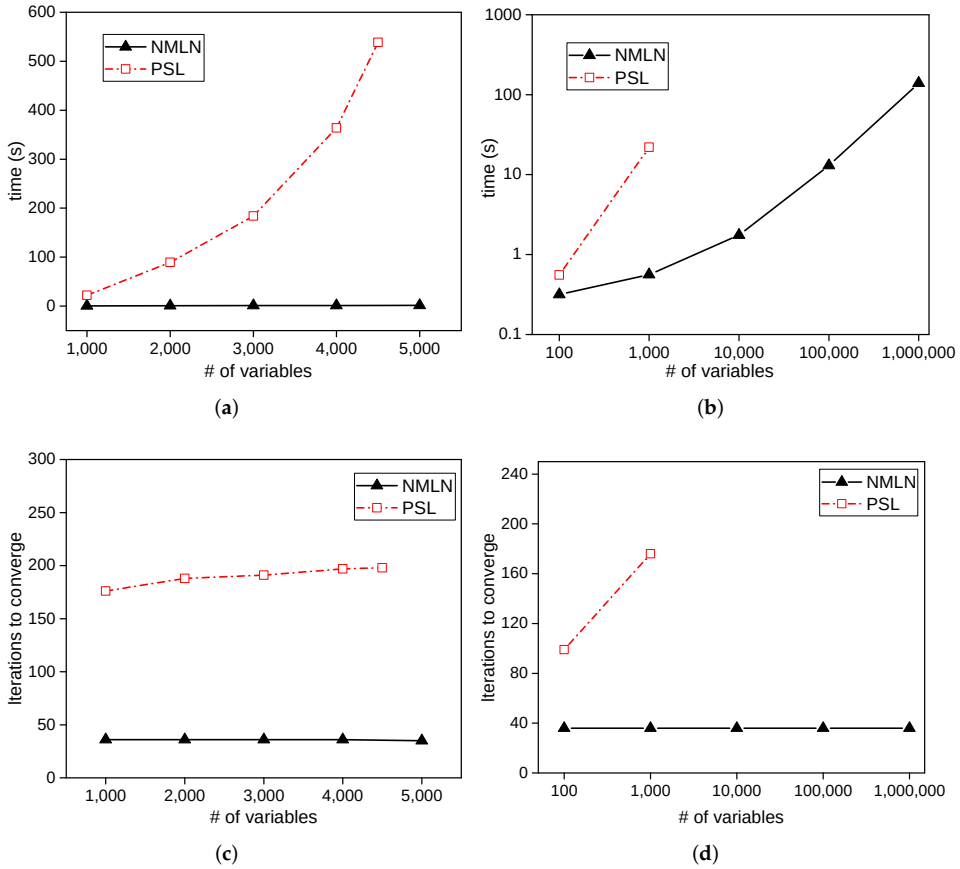


Figure 5. Scalability evaluation on NMLN vs. PSL: (**a**) runtime; (**b**) runtime log scale; (**c**) iterations to converge; (**d**) iteration log scale.

6.3. Sensitivity Evaluation

In this subsection, we evaluate the performance sensitivity of the NMLN w.r.t. the number of parallel threads, the base of exponential function α, and the step size ρ. In our empirical study, except the evaluated parameter, all the other parameters were set to the same values. We ran the evaluation of parallel threads on the synthetic data for scalability evaluation, since the size of the variables has a significant impact on the parallel methods. For evaluations on the parameters of α and ρ, we only present the evaluation results on the original mobile phone rank data due to the reason that different sizes of variables seem to have no effect on the results.

The evaluation results on the number of parallel threads are presented in Figure 6, in which the x-axis denotes the number of variables and the y-axis denotes the percentage of runtime compared with the runtime of non-paralleled method ($Threads = 1$) spent on the same data as the baseline. It can be observed that the runtime of paralleled inference decreases significantly when the size of the variables is large. Specifically, when the number of threads is set to six, the runtime of inference decreases to 23% and 27%, respectively, on 1000 K and 100 K variables. However, if the variables are smaller than 1K, the runtime decreases only marginally with the increase of the threads. This should not be surprising because small tasks are not suitable for parallelization.

The evaluation results on the base of exponential function α are presented in Figure 7, in which the parameter varies from e to 100. It can be observed that the performance of the NMLN fluctuates only marginally within a long range of α for both the primal residual and dual residual. Therefore, the NMLN inference is stable to take different base values.

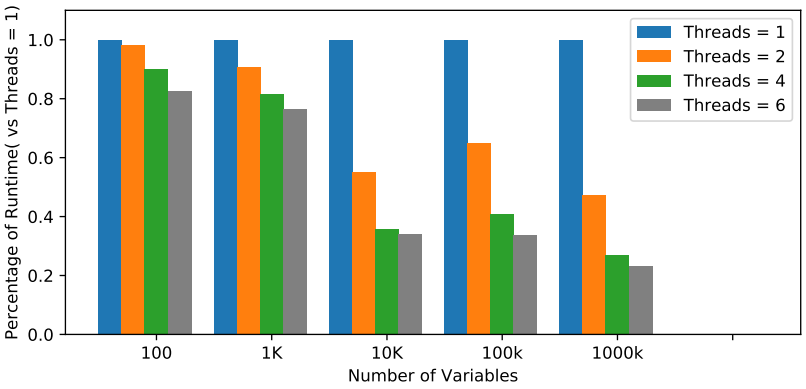


Figure 6. Sensitivity evaluation w.r.t. the number of parallel threads.

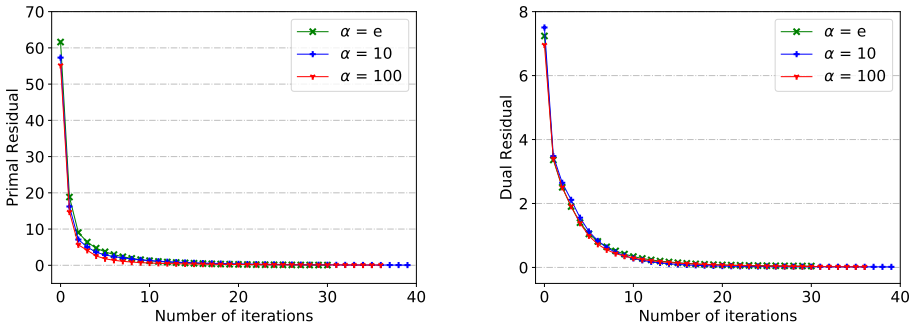


Figure 7. Sensitivity evaluation w.r.t. α: (**a**) primal residual; (**b**) dual residual.

The evaluation results on the step size ρ are presented in Figure 8, in which the parameter varies from 0.1 to 1.0. It can be observed that the larger value of ρ leads to a faster convergence speed on the primal residual and a slower convergence speed on the dual residual. Thus, the step size ρ should be set to a proper value (0.5) to balance the two conditions.

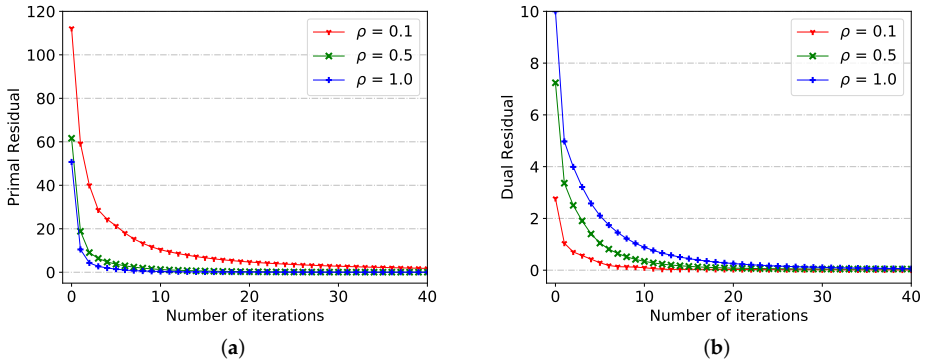


Figure 8. Sensitivity evaluation w.r.t. ρ: (**a**)primal residual; (**b**) dual residual.

7. Conclusions

Current MLN solutions cannot support knowledge inference involving arithmetic expressions. In this paper, we propose the Numerical Markov Logic Network (NMLN) to enable effective and efficient inference of hybrid knowledge involving both logic and arithmetic expressions. We define the exp-loss function as the metric to integrate arithmetic inequalities and logic formulas. By exploiting the decomposition of exp-loss functions, our method reduces the computational complexity from $O\left(\theta n^k\right)$ to $O(\theta nk)$, such that

the inference has a good scalability for the issue of clause explosion. We also present a parallel solution for hybrid knowledge inference based on convex optimization. The proposed approach can achieve better prediction accuracy while significantly reducing the inference time.

Author Contributions: Conceptualization, P.Z. and Q.C.; methodology, P.Z.; software, P.Z.; validation, P.Z., Q.C. and B.H.; formal analysis, Q.C.; investigation, Q.C.; resources, Z.L.; data curation, B.H.; writing—original draft preparation, P.Z.; writing—review and editing, Q.C. and M.A.; visualization, P.Z.; supervision, Q.C.; project administration, Z.L.; funding acquisition, Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: The work was supported by the National Key Research and Development Program of China (2018YFB1003400), National Natural Science Foundation of China (61732014, 61672432), Fundamental Research Funds for the Central Universities (Program No. 3102019DX1004) and Natural Science Basic Research Plan in Shaanxi Province of China (Program No. 2018JM6086).

Data Availability Statement: Mobile phone data at https://benchmarks.ul.com/ (accessed on 3 June 2018), AIDA-CONLL data at https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/aida/downloads/ (accessed on 30 November 2018), Wiki-Sport and Wiki-FourDomains data at http://en.wikipedia.org/wiki/Wikipedia:Featured_articles (accessed on 10 January 2021).

Conflicts of Interest: We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, there is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled "Numerical Markov Logic Network: A Scalable Probabilistic Framework for Hybrid Knowledge Inference", which has been approved by all authors. I would like to declare on behalf of my co-authors that the work described is original research that has not been submitted or published in other journals previously, and not under consideration for publication elsewhere, in whole or in part.

Appendix A. Knowledge Rules in the Phone Dataset

Table A1. Knowledge rules in the phone dataset.

	Knowledge Rules	Weight
r_1	$core(c_1) \geqslant core(c_2) \rightarrow fastcpu(c_1) \geqslant fastcpu(c_2)$	2
r_2	$frequency(c_1) \geqslant frequency(c_2) \rightarrow fastcpu(c_1) \geqslant fastcpu(c_2)$	4
r_3	$sccore(c_1) \geqslant sccore(c_2) \rightarrow fastcpu(c_1) \geqslant fastcpu(c_2)$	1
r_4	$secfrequency(c_1) \geqslant secfrequency(c_2) \rightarrow fastcpu(c_1) \geqslant fastcpu(c_2)$	2
r_5	$fastcpu(c_1) \geqslant fastcpu(c_2)\&hascpu(p_1, c_1)\&hascpu(p_2, c_2) \rightarrow performance(p_1) \geqslant performance(p_2)$	1
r_6	$memory(p_1) \geqslant memory(p_2) \rightarrow performance(p_1) \geqslant performance(p_2)$	1

Appendix B. Knowledge Rules in the Aida Dataset

We show the meaning of all relations in the rules as follows:

- $\mathbf{prior}(\mathbf{m}, \mathbf{e})$: the prior distribution computed by the number of entities linking to e.
- $\mathbf{anchormisim}(\mathbf{m}, \mathbf{e})$: the similarity measured by mutual information according to the "hasWikipediaAnchorText" file in YAGO2.
- $\mathbf{anchormisim}(\mathbf{m}, \mathbf{e})$: the similarity measured by word2vector according to the "hasWikipediaAnchorText" file in YAGO2.
- $\mathbf{catemisim}(\mathbf{m}, \mathbf{e})$: the similarity measured by mutual information according to the "hasWikipediaCategory" file in YAGO2.

- **catewvsim**$(\mathbf{m}, \mathbf{e})$: the similarity measured by word2vector according to the "hasWikipediaCategory" file in YAGO2.
- **distance**$(\mathbf{m}, \mathbf{e})$: the edit distance-based similarity.
- **synsim**$(\mathbf{m}, \mathbf{e})$: the syntactical similarity computed by WordNet and YAGO2.
- **coherence**$(\mathbf{m}, \mathbf{e})$: we find the max candidate of $(mention, entity)$ pairs for one document, for which all entities in the set have the maximum word similarity.

Table A2. Knowledge rules in the Aida dataset.

	Knowledge Rules	Weight
r_1	$anchormisim(m,e_1) - anchormisim(m,e_2) \geqslant 0.008 \& prior(m,e_1) - prior(m,e_2) \geqslant 0.033 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	1.11
r_2	$distance(m,e_1) - distance(m,e_2) \geqslant -0.007 \& prior(m,e_1) - prior(m,e_2) \geqslant -0.016 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	0.75
r_3	$anchormisim(m,e_1) - anchormisim(m,e_2) \leqslant -0.008 \& prior(m,e_1) - prior(m,e_2) \leqslant 0.002 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	0.95
r_4	$anchorwvsim(m,e_1) - anchorwvsim(m,e_2) \leqslant -0.128 \& prior(m,e_1) - prior(m,e_2) \leqslant 0.108 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	0.77
r_5	$prior(m,e_1) - prior(m,e_2) \leqslant -0.178 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	0.95
r_6	$catewvsim(m,e_1) - catewvsim(m,e_2) \geqslant -0.811 \& prior(m,e_1) - prior(m,e_2) \geqslant -0.006 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	0.99
r_7	$anchorwvsim(m,e_1) - anchorwvsim(m,e_2) \leqslant 0.436 \& catewvsim(m,e_1) - catewvsim(m,e_2) \leqslant 0.797 \& prior(m,e_1) - prior(m,e_2) \leqslant -0.006 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	0.74
r_8	$anchormisim(m,e_1) - anchormisim(m,e_2) \leqslant -0.383 \& catemisim(m,e_1) - catemisim(m,e_2) \leqslant -0.0 \& prior(m,e_1) - prior(m,e_2) \leqslant 0.032 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	2.26
r_9	$anchormisim(m,e_1) - anchormisim(m,e_2) \geqslant 0.006 \& anchorwvsim(m,e_1) - anchorwvsim(m,e_2) \geqslant 0.445 \& catewvsim(m,e_1) - catewvsim(m,e_2) \geqslant -0.193 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	0.89
r_{10}	$anchormisim(m,e_1) - anchormisim(m,e_2) \leqslant 0.006 \& catewvsim(m,e_1) - catewvsim(m,e_2) \geqslant -0.057 \& distance(m,e_1) - distance(m,e_2) \geqslant 0.41 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	0.73
r_{11}	$anchormisim(m,e_1) - anchormisim(m,e_2) \geqslant -0.008 \& distance(m,e_1) - distance(m,e_2) \geqslant -0.007 \& prior(m,e_1) - prior(m,e_2) \geqslant -0.016 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	1.97

Table A2. *Cont.*

	Knowledge Rules	Weight
r_{12}	$anchormisim(m,e_1) - anchormisim(m,e_2) \leqslant -0.013 \& distance(m,e_1) - distance(m,e_2) \leqslant -0.01 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	0.9
r_{13}	$anchormisim(m,e_1) - anchormisim(m,e_2) \leqslant 0.006 \& catewvsim(m,e_1) - catewvsim(m,e_2) \geqslant -0.057 \& distance(m,e_1) - distance(m,e_2) \geqslant 0.41 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	0.73
r_{14}	$anchormisim(m,e_1) - anchormisim(m,e_2) \geqslant -0.006 \& catemisim(m,e_1) - catemisim(m,e_2) \geqslant -0.172 \& distance(m,e_1) - distance(m,e_2) \geqslant 0.007 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	0.95
r_{15}	$anchormisim(m,e_1) - anchormisim(m,e_2) \leqslant 0.092 \& catemisim(m,e_1) - catemisim(m,e_2) \leqslant 0.0 \& distance(m,e_1) - distance(m,e_2) \leqslant 0.007 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	0.95
r_{16}	$catewvsim(m,e_1) - catewvsim(m,e_2) \leqslant 0.811 \& prior(m,e_1) - prior(m,e_2) \leqslant -0.173 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	1.21
r_{17}	$anchormisim(m,e_1) - anchormisim(m,e_2) \geqslant 0.006 \& catemisim(m,e_1) - catemisim(m,e_2) \geqslant -0.189 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	0.82
r_{18}	$anchorwvsim(m,e_1) - anchorwvsim(m,e_2) \geqslant 0.131 \& distance(m,e_1) - distance(m,e_2) \geqslant -0.354 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	0.84
r_{19}	$anchorwvsim(m,e_1) - anchorwvsim(m,e_2) \leqslant 0.809 \& anchorwvsim(m,e_1) - anchorwvsim(m,e_2) \geqslant 0.131 \& prior(m,e_1) - prior(m,e_2) \geqslant -0.094 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	1.04
r_{20}	$anchormisim(m,e_1) - anchormisim(m,e_2) \geqslant 0.008 \& catewvsim(m,e_1) - catewvsim(m,e_2) \geqslant -0.984 \& prior(m,e_1) - prior(m,e_2) \geqslant -0.003 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	1.58
r_{21}	$anchormisim(m,e_1) - anchormisim(m,e_2) \leqslant 0.01 \& anchormisim(m,e_1) - anchormisim(m,e_2) \geqslant 0.008 \& catemisim(m,e_1) - catemisim(m,e_2) \leqslant -0.172 \& prior(m,e_1) - prior(m,e_2) \leqslant -0.003 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	5.0
r_{22}	$anchorwvsim(m,e_1) - anchorwvsim(m,e_2) \leqslant -0.185 \& catemisim(m,e_1) - catemisim(m,e_2) \leqslant 0.061 \& distance(m,e_1) - distance(m,e_2) \leqslant -0.007 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	1.59 :
r_{23}	$anchorwvsim(m,e_1) - anchorwvsim(m,e_2) \geqslant 0.09 \& catewvsim(m,e_1) - catewvsim(m,e_2) \geqslant -0.095 \& distance(m,e_1) - distance(m,e_2) \geqslant -0.389 \& prior(m,e_1) - prior(m,e_2) \geqslant -0.124 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	1.69

Table A2. *Cont.*

	Knowledge Rules	Weight
r_{24}	$anchorwvsim(m,e_1) - anchorwvsim(m,e_2) \leqslant 0.09 \& distance(m,e_1) - distance(m,e_2) \geqslant 0.272 \& prior(m,e_1) - prior(m,e_2) \geqslant -0.007 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	1.3
r_{25}	$anchorwvsim(m,e_1) - anchorwvsim(m,e_2) \leqslant 0.093 \& distance(m,e_1) - distance(m,e_2) \leqslant -0.298 \& prior(m,e_1) - prior(m,e_2) \leqslant 0.077 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	3.29
r_{26}	$anchormisim(m,e_1) - anchormisim(m,e_2) \geqslant -0.008 \& catemisim(m,e_1) - catemisim(m,e_2) \geqslant -0.0 \& catewvsim(m,e_1) - catewvsim(m,e_2) \geqslant 0.163 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	3.28
r_{27}	$anchorwvsim(m,e_1) - anchorwvsim(m,e_2) \leqslant 0.229 \& catemisim(m,e_1) - catemisim(m,e_2) \leqslant -0.0 \& prior(m,e_1) - prior(m,e_2) \leqslant -0.012 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	1.43
r_{28}	$anchormisim(m,e_1) - anchormisim(m,e_2) \geqslant -0.031 \& distance(m,e_1) - distance(m,e_2) \geqslant -0.007 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	5.0
r_{29}	$anchormisim(m,e_1) - anchormisim(m,e_2) \leqslant 0.006 \& anchorwvsim(m,e_1) - anchorwvsim(m,e_2) \geqslant 0.093 \& distance(m,e_1) - distance(m,e_2) \leqslant -0.007 \& synsim(m,e_1) - synsim(m,e_2) \leqslant -0.158 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	5.0
r_{30}	$anchormisim(m,e_1) - anchormisim(m,e_2) \geqslant$- $0.119 \& catemisim(m,e_1) - catemisim(m,e_2) \geqslant -0.011 \& catewvsim(m,e_1) - catewvsim(m,e_2) \geqslant -0.816 \& prior(m,e_1) - prior(m,e_2) \geqslant -0.006 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	1.58
r_{31}	$anchormisim(m,e_1) - anchormisim(m,e_2) \leqslant -0.893 \& distance(m,e_1) - distance(m,e_2) \geqslant 0.397 \& prior(m,e_1) - prior(m,e_2) \leqslant -0.006 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	5.0
r_{32}	$anchormisim(m,e_1) - anchormisim(m,e_2) \leqslant 0.119 \& catemisim(m,e_1) - catemisim(m,e_2) \leqslant 0.15 \& distance(m,e_1) - distance(m,e_2) \leqslant 0.397 \& prior(m,e_1) - prior(m,e_2) \leqslant -0.006 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	1.28
r_{33}	$catewvsim(m,e_1) - catewvsim(m,e_2) \geqslant -0.796 \& distance(m,e_1) - distance(m,e_2) \geqslant -0.573 \& prior(m,e_1) - prior(m,e_2) \geqslant 0.003 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	1.15
r_{34}	$catewvsim(m,e_1) - catewvsim(m,e_2) \leqslant 0.804 \& distance(m,e_1) - distance(m,e_2) \leqslant 0.397 \& prior(m,e_1) - prior(m,e_2) \leqslant -0.085 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	1.3
r_{35}	$catemisim(m,e_1) - catemisim(m,e_2) \geqslant -0.17 \& catewvsim(m,e_1) - catewvsim(m,e_2) \geqslant -0.807 \& distance(m,e_1) - distance(m,e_2) \geqslant 0.006 \& prior(m,e_1) - prior(m,e_2) \geqslant -0.033 \rightarrow link(m,e_1) \geqslant link(m,e_2)$	1.33

Table A2. *Cont.*

	Knowledge Rules	Weight
r_{36}	$catemisim(m,e_1) - catemisim(m,e_2) \geqslant -0.012 \& catewvsim(m,e_1) - catewvsim(m,e_2) \leqslant -0.99 \& distance(m,e_1) - distance(m,e_2) \leqslant 0.006 \& prior(m,e_1) - prior(m,e_2) \geqslant 0.032 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	0.96
r_{37}	$anchormisim(m,e_1) - anchormisim(m,e_2) \leqslant 0.362 \& distance(m,e_1) - distance(m,e_2) \leqslant 0.006 \& prior(m,e_1) - prior(m,e_2) \leqslant -0.162 \rightarrow link(m,e_1) \leqslant link(m,e_2)$	2.88
r_{38}	$coherence(m,e) \rightarrow link(m,e)$	1
r_{39}	$link(m,e) \leqslant 0.2$	1

References

1. Richardson, M.; Domingos, P.M. Markov logic networks. *Mach. Learn.* **2006**, *62*, 107–136. [CrossRef]
2. Banerjee, O.; El Ghaoui, L.; d'Aspremont, A. Model Selection Through Sparse Maximum Likelihood Estimation for Multivariate Gaussian or Binary Data. *J. Mach. Learn. Res.* **2008**, *9*, 485–516.
3. Dong, X.L.; Gabrilovich, E.; Heitz, G.; Horn, W.; Murphy, K.; Sun, S.; Zhang, W. From Data Fusion to Knowledge Fusion. *PVLDB* **2014**, *7*, 881–892. [CrossRef]
4. Jiang, S.; Lowd, D.; Dou, D. Learning to Refine an Automatically Extracted Knowledge Base Using Markov Logic. In Proceedings of the 12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, 10–13 December 2012; pp. 912–917. [CrossRef]
5. Zhang, C.; Ré, C.; Cafarella, M.J.; Shin, J.; Wang, F.; Wu, S. DeepDive: declarative knowledge base construction. *Commun. ACM* **2017**, *60*, 93–102. [CrossRef]
6. Singla, P.; Domingos, P.M. Entity Resolution with Markov Logic. In Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), Hong Kong, China, 18–22 December 2006; pp. 572–582. [CrossRef]
7. Niu, F.; Ré, C.; Doan, A.; Shavlik, J.W. Tuffy: Scaling up Statistical Inference in Markov Logic Networks using an RDBMS. *PVLDB* **2011**, *4*, 373–384. [CrossRef]
8. Chen, Y.; Wang, D.Z. Knowledge expansion over probabilistic knowledge bases. In Proceedings of the International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, 22–27 June 2014; pp. 649–660. [CrossRef]
9. Sa, C.D.; Ratner, A.; Ré, C.; Shin, J.; Wang, F.; Wu, S.; Zhang, C. Incremental knowledge base construction using DeepDive. *VLDB J.* **2017**, *26*, 81–105. [CrossRef]
10. Bach, S.H.; Broecheler, M.; Huang, B.; Getoor, L. Hinge-Loss Markov Random Fields and Probabilistic Soft Logic. *J. Mach. Learn. Res.* **2017**, *18*, 109:1–109:67.
11. Wick, M.L.; McCallum, A.; Miklau, G. Scalable Probabilistic Databases with Factor Graphs and MCMC. *PVLDB* **2010**, *3*, 794–804. [CrossRef]
12. Zhang, C.; Ré, C. Towards High-throughput Gibbs Sampling at Scale: A Study Across Storage Managers. In Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, ACM, SIGMOD '13, New York, NY, USA, 22–27 June 2013; pp. 397–408. [CrossRef]
13. Krapu, C.; Borsuk, M. Probabilistic programming: A review for environmental modellers. *Environ. Model. Softw.* **2019**, *114*, 40–48. [CrossRef]
14. Salvatier, J.; Wiecki, T.V.; Fonnesbeck, C.; Elkan, C. Probabilistic programming in Python using PyMC3. *PeerJ Comput. Sci.* **2016**, *2*. [CrossRef]
15. Tran, D.; Kucukelbir, A.; Dieng, A.B.; Rudolph, M.R.; Liang, D.; Blei, D.M. Edward: A library for probabilistic modeling, inference, and criticism. *arXiv* **2016**, arXiv:1610.09787.
16. Bingham, E.; Chen, J.P.; Jankowiak, M.; Obermeyer, F.; Pradhan, N.; Karaletsos, T.; Singh, R.; Szerlip, P.A.; Horsfall, P.; Goodman, N.D. Pyro: Deep Universal Probabilistic Programming. *J. Mach. Learn. Res.* **2019**, *20*, 28:1–28:6.
17. Zhong, P.; Li, Z.; Chen, Q.; Wang, Y.; Wang, L.; Ahmed, M.H.M.; Fan, F. POOLSIDE: An Online Probabilistic Knowledge Base for Shopping Decision Support. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, 6–10 November 2017; pp. 2559–2562. [CrossRef]
18. Gutiérrez-Basulto, V.; Jung, J.C.; Kuzelka, O. Quantified Markov Logic Networks. In Proceedings of the Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference, KR 2018, Tempe, Arizona, 30 October–2 November 2018; pp. 602–612.
19. Sabek, I.; Musleh, M.; Mokbel, M.F. Flash in Action: Scalable Spatial Data Analysis Using Markov Logic Networks. *PVLDB* **2019**, *12*, 1834–1837. [CrossRef]

20. Gayathri, K.; Easwarakumar, K.; Elias, S. Probabilistic ontology based activity recognition in smart homes using Markov Logic Network. *Knowl.-Based Syst.* **2017**, *121*, 173–184. [CrossRef]
21. Schoenfisch, J.; Meilicke, C.; von Stülpnagel, J.; Ortmann, J.; Stuckenschmidt, H. Root cause analysis in IT infrastructures using ontologies and abduction in Markov Logic Networks. *Inf. Syst.* **2018**, *74*, 103–116. [CrossRef]
22. Kennington, C.; Schlangen, D. Situated incremental natural language understanding using Markov Logic Networks. *Comput. Speech Lang.* **2014**, *28*, 240–255. [CrossRef]
23. Ge, C.; Gao, Y.; Miao, X.; Yao, B.; Wang, H. A Hybrid Data Cleaning Framework Using Markov Logic Networks. *IEEE Trans. Knowl. Data Eng.* **2020**, 1. [CrossRef]
24. Sabek, I. Adopting Markov Logic Networks for Big Spatial Data and Applications. In Proceedings of the International Conference on Very Large Data Bases (VLDB), Los Angeles, CA, USA, 26–30 August 2019.
25. Hao, W.; Menglin, J.; Guohui, T.; Qing, M.; Guoliang, L. R-KG: A Novel Method for Implementing a Robot Intelligent Service. *AI* **2020**, *1*, 117–140. [CrossRef]
26. Sarkhel, S.; Venugopal, D.; Singla, P.; Gogate, V. Lifted MAP Inference for Markov Logic Networks. In Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, 22–25 April 2014; pp. 859–867.
27. Beedkar, K.; Corro, L.D.; Gemulla, R. Fully Parallel Inference in Markov Logic Networks. In Proceedings of the Datenbanksysteme für Business, Technologie und Web (BTW), 15. Fachtagung des GI-Fachbereichs "Datenbanken und Informationssysteme" (DBIS), Magdeburg, Germany, 11–15 March 2013; pp. 205–224.
28. Zhou, X.; Chen, Y.; Wang, D.Z. ArchimedesOne: Query Processing over Probabilistic Knowledge Bases. *PVLDB* **2016**, *9*, 1461–1464. [CrossRef]
29. Sun, Z.; Zhao, Y.; Wei, Z.; Zhang, W.; Wang, J. Scalable learning and inference in Markov logic networks. *Int. J. Approx. Reason.* **2017**, *82*, 39–55. [CrossRef]
30. Singla, P.; Domingos, P.M. Discriminative Training of Markov Logic Networks. In Proceedings of the Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, Pittsburgh, PA, USA, 9–13 July 2005; pp. 868–873.
31. Lowd, D.; Domingos, P.M. Efficient Weight Learning for Markov Logic Networks. In Proceedings of the Knowledge Discovery in Databases: PKDD 2007, 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, 17–21 September 2007; pp. 200–211. [CrossRef]
32. Huynh, T.N.; Mooney, R.J. Max-Margin Weight Learning for Markov Logic Networks. In Proceedings of the Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2009, Bled, Slovenia, 7–11 September 2009; pp. 564–579. [CrossRef]
33. Kok, S.; Domingos, P.M. Learning the structure of Markov logic networks. In Proceedings of the Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, 7–11 August 2005; pp. 441–448. [CrossRef]
34. Mihalkova, L.; Mooney, R.J. Bottom-up learning of Markov logic network structure. In Proceedings of the Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, OR, USA, 20–24 June 2007; pp. 625–632. [CrossRef]
35. Khot, T.; Natarajan, S.; Kersting, K.; Shavlik, J. Gradient-based boosting for statistical relational learning: the Markov logic network and missing data cases. *Mach. Learn.* **2015**, *100*. [CrossRef]
36. Marra, G.; Kuzelka, O. Neural Markov Logic Networks. *arXiv* **2019**, arXiv:1905.13462.
37. Wang, J.; Domingos, P.M. Hybrid Markov Logic Networks. In Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, IL, USA, 13–17 July 2008; pp. 1106–1111.
38. Klir, G.J.; Yuan, B. *Fuzzy Sets and Fuzzy Logic—Theory and Applications*; Prentice Hall: Upper Saddle River, NJ, USA, 1995.
39. Boyd, S.P.; Parikh, N.; Chu, E.; Peleato, B.; Eckstein, J. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Found. Trends Mach. Learn.* **2011**, *3*, 1–122. [CrossRef]
40. Sang, E.F.T.K.; Meulder, F.D. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In Proceedings of the Seventh Conference on Natural Language Learning, CoNLL 2003, Edmonton, AB, Canada, 31 May–1 June 2003; pp. 142–147.
41. Hoffart, J.; Suchanek, F.M.; Berberich, K.; Lewis-Kelham, E.; de Melo, G.; Weikum, G. YAGO2: Exploring and querying world knowledge in time, space, context, and many languages. In Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, 28 March–1 April 2011; pp. 229–232. [CrossRef]