Knowledge Base Completion Using Embeddings and Rules

Quan Wang, Bin Wang, Li Guo
Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{wangquan,wangbin,guoli}@iie.ac.cn

Abstract

Knowledge bases (KBs) are often greatly incomplete, necessitating a demand for KB completion. A promising approach is to embed KBs into latent spaces and make inferences by learning and operating on latent representations. Such embedding models, however, do not make use of any rules during inference and hence have limited accuracy. This paper proposes a novel approach which incorporates rules seamlessly into embedding models for KB completion. It formulates inference as an integer linear programming (ILP) problem, with the objective function generated from embedding models and the constraints translated from rules. Solving the ILP problem results in a number of facts which 1) are the most preferred by the embedding models, and 2) comply with all the rules. By incorporating rules, our approach can greatly reduce the solution space and significantly improve the inference accuracy of embedding models. We further provide a slacking technique to handle noise in KBs, by explicitly modeling the noise with slack variables. Experimental results on two publicly available data sets show that our approach significantly and consistently outperforms state-of-the-art embedding models in KB completion. Moreover, the slacking technique is effective in identifying erroneous facts and ambiguous entities, with a precision higher than 90%.

1 Introduction

Knowledge bases (KBs), e.g., WordNet [Miller, 1995], Freebase [Bollacker *et al.*, 2008], and YAGO [Suchanek *et al.*, 2007], are extremely useful resources for many AI related applications. KBs contain rich information of entities and their relations, stored in triples of the form (*head entity*, *relation*, *tail entity*), called facts. Although typical KBs may contain millions or even billions of facts, they are still greatly incomplete [West *et al.*, 2014]. KB completion, i.e., automatically inferring missing facts from existing ones, has thus become an increasingly important task.

A promising approach to KB completion is to embed KBs into low-dimensional vector spaces [Bordes *et al.*, 2011; 2013; 2014; Nickel *et al.*, 2011; Socher *et al.*, 2013]. Specifically, given a KB, entities and relations are first represented in a low-dimensional vector space, and for each triple, a scoring function is defined to measure its plausibility. Then, the representations of entities and relations (i.e. embeddings) are learned by maximizing the total plausibility of existing facts. For any missing fact, its plausibility can be predicted by using the learned embeddings. By learning and operating on latent representations, such embedding models are able to capture some unobservable but intrinsic properties of entities and relations [Jenatton *et al.*, 2012].

The main drawback of KB embedding models is their purely data-driven fashion. Most of the previous embedding models make inferences based solely on existing facts, utilizing neither physical nor logical rules. Here, logical rules are those involving logic and deduction (e.g., entities linked by the relation `HasWife` should also be linked by the relation `HasSpouse`); physical rules refer to those non-logical ones enforcing physical restraints (e.g., both arguments of the relation `HasSpouse` need to be `Person` entities). Rules have been demonstrated to play a pivotal role in inference [Jiang *et al.*, 2012; Pujara *et al.*, 2013], and hence are of critical importance to KB completion.

In this paper, we propose a novel KB completion approach that infers missing facts using both embeddings and rules. The new approach formulates inference as a constrained maximization problem, or more specifically, an integer linear programming (ILP) problem. The objective function is the aggregated plausibility of all candidate facts, predicted by a specific KB embedding model; and the constraints are translated from physical and logical rules. Solving the optimization problem results in a set of facts which 1) have the highest plausibility predicted by the embedding model, and 2) comply with all the rules and hence are physically and logically favourable. Figure 1 sketches the approach.

The advantages of our approach are three-fold: 1) The use of rules greatly reduces the solution space and significantly enhances inference accuracy; 2) It naturally preserves the benefits of embedding models, capable of capturing intrinsic properties of data; 3) It is a general framework, applicable to a wide variety of embedding models and rules.

Moreover, given that KBs (especially those constructed by information extraction techniques) can be very noisy, we provide a simple, yet effective slacking technique to deal with

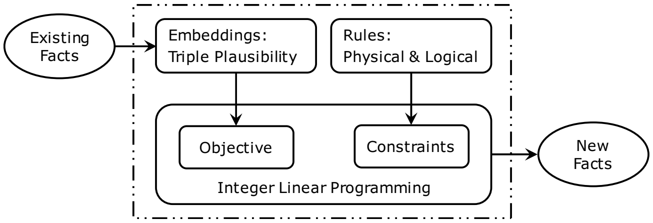
Existing Facts
Embeddings: Triple Plausibility
Rules: Physical & Logical
Objective
Constraints
Integer Linear Programming
New Facts

Figure 1: Overview of our approach.

the noise. The key idea is to explicitly model the noise with slack variables, allowing the observed triples to be false.

We evaluate our approach on publicly available data created using NELL [Carlson *et al.*, 2010]. Experimental results show that by incorporating rules, our approach significantly and consistently outperforms state-of-the-art KB embedding models. Furthermore, the slacking technique is effective in identifying erroneous facts and ambiguous entities, with a precision higher than 90%.

2 Related Work

KB completion is to automatically infer missing facts based on existing ones in a KB. The literature falls into three major categories: 1) methods based on embedding strategies that model connections in a KB as resulting from latent factors (e.g. [Nickel *et al.*, 2011; Bordes *et al.*, 2013]); 2) methods based on Markov random fields (MRFs) that make inferences via first-order logic (e.g. [Jiang *et al.*, 2012]) or probabilistic soft logic (e.g. [Pujara *et al.*, 2013]); 3) methods based on path ranking algorithms that search entity connections using random walks (e.g. [Lao *et al.*, 2011; Dong *et al.*, 2014]). This paper focuses on the first category.

The key idea of embedding-based KB completion is to represent entities and relations in latent spaces and model candidate facts as resulting from latent factors. RESCAL [Nickel *et al.*, 2011] and TransE [Bordes *et al.*, 2013] are two typical methods, learning latent representations by minimizing a reconstruction loss or a margin-based ranking loss respectively. Other approaches include [Bordes *et al.*, 2011; 2014; Socher *et al.*, 2013; Wang *et al.*, 2014; Lin *et al.*, 2015; Guo *et al.*, 2015]. During learning and inference, these embedding models only exploit existing facts, and do not make use of rules. TRESCAL [Chang *et al.*, 2014] tries to encode rules into RESCAL. However, it focuses solely on a single rule (i.e., arguments of a relation should be entities of certain types). Rocktäschel et al. [2014] recently proposed to embed first-order logic into low-dimensional vector spaces. But in their work rules are not imposed directly for inference and cannot explicitly reduce the solution space. Our work differs in that we provide a general framework oriented towards inference, capable of incorporating various types of rules.

Rules, particularly logical rules, have been studied extensively in MRF-based KB completion approaches, represented in first-order logic [Richardson and Domingos, 2006] or probabilistic soft logic [Bröcheler *et al.*, 2010]. This paper incorporates rules into embedding models, represented as constraints of a maximization problem. Moreover, besides logical rules, physical rules are also investigated in our approach.

Integer linear programming (ILP) refers to constrained optimization where both the objective and the constraints are linear equations with integer variables. It has been widely applied in many different areas, e.g., AI planning [Vossen *et al.*, 1999; Do *et al.*, 2007], natural language processing [Roth and Yih, 2004; Riedel and Clarke, 2006], and computational biology [Dittrich *et al.*, 2008; Wang and Xu, 2013]. This paper employs ILP to integrate embedding models and rules in a unified framework for KB completion.

3 Our Approach

As illustrated in Figure 1, our approach consists of three key components: 1) We employ KB embedding models to predict the plausibility of each candidate fact; 2) We introduce physical and logical rules to impose restraints on candidate facts; 3) We integrate the first two components by ILP, with the objective function generated from the embedding models and the constraints translated from the rules. Facts inferred in this way will have the highest plausibility predicted by KB embedding, and at the same time comply with all the rules.

3.1 Embedding Knowledge Bases

Suppose we are given a KB consisting of n entities and m relations. The facts observed are stored as a set of triples $\mathcal{O} = \{(e_i, r_k, e_j)\}$. A triple (e_i, r_k, e_j) indicates that entity e_i and entity e_j are connected by relation r_k. KB embedding aims to 1) embed the entities and relations into a latent space, and 2) model and predict the plausibility of each candidate fact in that space. We employ three KB embedding models: RESCAL [Nickel *et al.*, 2011], TRESCAL [Chang *et al.*, 2014], and TransE [Bordes *et al.*, 2013].

RESCAL represents each entity e_i as a vector $\mathbf{e}_i \in \mathbb{R}^d$ and each relation r_k as a matrix $\mathbf{R}_k \in \mathbb{R}^{d \times d}$ in the latent space. Given a triple (e_i, r_k, e_j), a bilinear scoring function

$$f\left(e_i, r_k, e_j\right) = \mathbf{e}_i^T \mathbf{R}_k \mathbf{e}_j$$

is used to model the plausibility of the triple. $\{\mathbf{e}_i\}$ and $\{\mathbf{R}_k\}$ are learned by minimizing a reconstruction loss, i.e.,

$$\min_{\{\mathbf{e}_i\},\{\mathbf{R}_k\}} \sum_k \sum_i \sum_j \left(y_{ij}^{(k)} - f\left(e_i, r_k, e_j\right)\right)^2 + \lambda \mathcal{R},$$

where $y_{ij}^{(k)}$ equals one if the triple $(e_i, r_k, e_j) \in \mathcal{O}$ and zero otherwise, and $\mathcal{R} = \sum_i \|\mathbf{e}_i\|_2^2 + \sum_k \|\mathbf{R}_k\|_F^2$ is a regularization term. An alternating least squares algorithm is adopted to solve the optimization problem.

TRESCAL is an extension of RESCAL, requiring the arguments of a relation to be entities of certain types. Given a relation r_k, let $\mathcal{H}_k$ and $\mathcal{T}_k$ be the sets of entities with compatible types (e.g., for the relation `CityLocatedInCountry`, $\mathcal{H}_k$ contains `City` entities and $\mathcal{T}_k$ `Country` entities). Learning is then conducted by reconstructing legitimate triples, i.e.,

$$\min_{\{\mathbf{e}_i\},\{\mathbf{R}_k\}} \sum_k \sum_{i \in \mathcal{H}_k} \sum_{j \in \mathcal{T}_k} \left(y_{ij}^{(k)} - f\left(e_i, r_k, e_j\right)\right)^2 + \lambda \mathcal{R},$$

solved again by an alternating least squares algorithm.[1]

[1] Both RESCAL and TRESCAL are originally presented in matrix forms.

TransE represents both entities and relations as vectors in the latent space. Given a triple (e_i, r_k, e_j) and the associated embeddings $\mathbf{e}_i, \mathbf{e}_j, \mathbf{r}_k \in \mathbb{R}^d$, TransE uses the following scoring function to measure the plausibility of the triple:

$$f(e_i, r_k, e_j) = \|\mathbf{e}_i + \mathbf{r}_k - \mathbf{e}_j\|_1 .$$

$\{\mathbf{e}_i\}$ and $\{\mathbf{r}_k\}$ are learned by minimizing a margin-based ranking loss, enforcing positive (i.e. observed) triples to have higher plausibility than negative ones, i.e.,

$$\min_{\{\mathbf{e}_i\},\{\mathbf{r}_k\}} \sum_{t^+ \in \mathcal{O}} \sum_{t^- \in \mathcal{N}_{t^+}} \left[\gamma - f(e_i, r_k, e_j) + f\left(e_i', r_k, e_j'\right)\right]_+ ,$$

where $t^+ = (e_i, r_k, e_j)$ is a positive triple; $\mathcal{N}_{t^+}$ contains negative triples constructed by replacing the entities in t^+; $t^- = \left(e_i', r_k, e_j'\right) \in \mathcal{N}_{t^+}$; and $\gamma > 0$ is a margin separating positive and negative triples.[2] Stochastic gradient descent is adopted to solve the optimization problem.

After we obtain the embeddings, for any missing triple, its plausibility can be predicted by using the scoring functions. Triples with higher plausibility are more likely to be true.

3.2 Imposing Physical & Logical Rules

We introduce three physical rules (Rule 1, 2, and 3) and a logical rule (Rule 4) to impose restraints on candidate facts.

Rule 1 (noisy observation). *Facts that already exist are very likely, but not necessarily to be true.* KBs, especially those constructed by information extraction techniques, can be very noisy. This rule is crucial in handling noisy KBs.

Rule 2 (argument type expectation). *Arguments of a relation should be entities of certain types.* For example, both arguments of the relation `HasSpouse` need to be `Person` entities. This rule is the same as considered in TRESCAL.

Rule 3 (at-most-one restraint). *For 1-To-Many/Many-To-1 relations, the first/second argument can take at most one entity; for 1-To-1 relations, both arguments can take at most one entity.* For example, `CityLocatedInCountry` is a Many-To-1 relation. Given any `City` entity, there exists at most one `Country` entity for which the resultant triple is true (i.e., a city locates in at most one country in the KB).

Rule 4 (simple implication). *Suppose relation r_1 implicates relation r_2, denoted as $r_1 \mapsto r_2$. Then, any two entities linked by r_1 should also be linked by r_2.* For example, `HasWife` $\mapsto$ `HasSpouse`.

These rules contain rich prior knowledge, and can greatly reduce the solution space during inference.

3.3 Integrating by Integer Linear Programming

We aggregate the above two components and formulate inference as an ILP problem. For each candidate fact (e_i, r_k, e_j), we use $w_{ij}^{(k)} = f(e_i, r_k, e_j)$ to represent the plausibility predicted by an embedding model, and introduce a Boolean decision variable $x_{ij}^{(k)}$ to indicate whether the fact is true or false. Our aim is then to find the best assignment to the decision variables, maximizing the overall plausibility and complying with all the rules. The ILP problem is given in Figure 2.[3] Here, t^+ is a fact already observed; $\mathcal{R}_{1-M}/\mathcal{R}_{M-1}/\mathcal{R}_{1-1}$ refers to the set of 1-To-Many/Many-To-1/1-To-1 relations; and the ID before each constraint indicates the corresponding rule (e.g., constraint R1 is translated from Rule 1).

$$\max_{\left\{x_{ij}^{(k)}, \epsilon_{ij}^{(k)}\right\}} \sum_k \sum_i \sum_j w_{ij}^{(k)} x_{ij}^{(k)} - \sum_{t^+ \in \mathcal{O}} \epsilon_{ij}^{(k)},$$

$$\begin{aligned}
\text{s.t.}\quad & \text{R1.} && x_{ij}^{(k)} + \epsilon_{ij}^{(k)} = 1,\ \forall t^+ \in \mathcal{O},\\
& \text{R2.} && x_{ij}^{(k)} = 0,\ \forall k, \forall i \notin \mathcal{H}_k, \forall j \notin \mathcal{T}_k,\\
& \text{R3.} && \textstyle\sum_i x_{ij}^{(k)} \le 1,\ \forall k \in \mathcal{R}_{1-M}, \forall j,\\
& \text{R3.} && \textstyle\sum_j x_{ij}^{(k)} \le 1,\ \forall k \in \mathcal{R}_{M-1}, \forall i,\\
& \text{R3.} && \textstyle\sum_i x_{ij}^{(k)} \le 1, \sum_j x_{ij}^{(k)} \le 1,\ \forall k \in \mathcal{R}_{1-1}, \forall i, \forall j,\\
& \text{R4.} && x_{ij}^{(k_1)} \le x_{ij}^{(k_2)},\ \forall r_{k_1} \mapsto r_{k_2}, \forall i, \forall j,\\
\text{where}\quad & && x_{ij}^{(k)} \in \{0,1\}, \forall k, i, j;\ \epsilon_{ij}^{(k)} \in \{0,1\}, \forall t^+ \in \mathcal{O}.
\end{aligned}$$

Figure 2: The associated ILP problem.

The most notable constraint is the first one, i.e., the noisy observation rule. We introduce a Boolean slack variable ϵ for each observed fact to explicitly model the noise: if $\epsilon = 1$, the observed fact ought to be false ($x = 0$). And we penalize the objective function with the noise, encouraging the observed facts to be true (but not necessarily).

Our approach has the following advantages: 1) By incorporating rules, it significantly improves the inference accuracy of existing KB embedding models; 2) The noisy observation rule (more specifically the slack variables) can automatically identify noise in KBs; 3) It is a general framework, applicable to a wide variety of embedding models, and capable of incorporating various types of rules.

4 Experiments

We conduct experiments to test the performance of our approach in 1) retrieving head/tail entities and 2) predicting new facts. We further investigate the effectiveness of the slack variables in 3) automatic noise detection.

4.1 Data Sets

We create two data sets Location and Sport using NELL, both containing five relations (listed in Table 1) and the associated triples. On the Sport data set, for the last two relations, only those triples related to "sport" are included. On both data sets, entities appearing only once are further removed, resulting in 195 entities and 231 triples on Location, and 477 entities and 710 triples on Sport. In NELL, the entity type information is encoded in a specific relation called `Generalization`. From this information, we obtain the argument type expectation for each relation, as suggested in [Chang *et al.*, 2014].

[2] Bordes et al. [2013] referred to the scoring function as an energy function, and required positive triples to have lower energies. The two formulations are equivalent.

[3] Constraint R4 might also be written as $x_{ij}^{(k_1)} \le x_{ji}^{(k_2)}$.

	Relation/Argument Type/Relation Type			Simple Implication Rules
Location	CityCapitalOfCountry	City/Country	1-To-1	CityCapitalOfCountry ↦ CityLocatedInCountry StateHasCapital ↦ CityLocatedInState
	CityLocatedInCountry	City/Country	M.-To-1	
	CityLocatedInState	City/State	M.-To-1	
	StateHasCapital	State/City	1-To-1	
	StateLocatedInCountry	State/Country	M.-To-1	
Sport	AthleteLedSportsTeam	Person/Sportteam	M.-To-1	AthleteLedSportsTeam ↦ AthletePlaysForTeam
	AthletePlaysForTeam	Person/Sportteam	M.-To-1	AthletePlaysForTeam ↦ PersonBelongsToOrganization
	CoachesTeam	Person/Sportteam	M.-To-1	CoachesTeam ↦ PersonBelongsToOrganization
	OrganizationHiredPerson	Sportteam/Person	1-To-M.	OrganizationHiredPerson ↦ PersonBelongsToOrganization
	PersonBelongsToOrganization	Person/Sportteam	M.-To-1	PersonBelongsToOrganization ↦ OrganizationHiredPerson

Table 1: Argument types, relation types, and simple implication rules on Location and Sport data sets.

To identify the relation type (i.e., 1-To-Many, Many-To-1, or 1-To-1), we follow the strategy proposed in [Bordes *et al.*, 2013]. For each relation, we compute the averaged number of heads e_i (tails e_j) appearing in the data set, given a tail e_j (head e_i). If the averaged number is smaller than 1.2, we label the argument as "1" and "Many" otherwise. We further create several simple implication rules on each data set. The argument types, relation types, and implication rules are given in Table 1. We will release the data upon request.

4.2 Retrieving Head/Tail Entities

This task is to complete a triple (e_i, r_k, e_j) with e_i or e_j missing, i.e., predict e_i given (r_k, e_j) or e_j given (e_i, r_k). It is called link prediction in previous work [Bordes *et al.*, 2011; 2013]. We test RESCAL, TRESCAL, and TransE in this task, before and after incorporating rules. A model with rules incorporated is denoted as r-RESCAL for example.

Evaluation protocol. To evaluate, for each data set, we split the triples into a training set and a test set, with the ratio of 4:1. The former is used for model training, and the latter for evaluation.[4] For each test triple, we replace the head/tail entity by each of the entities *with compatible types*, and rank the resultant corrupted triples in descending order, according to the plausibility (before incorporating rules) or the decision variables (after incorporating rules). Then we check whether the original correct triple ranks *first*. We corrupt head entities for 1-To-Many relations, tail entities for Many-To-1 relations, and both entities for 1-To-1 relations. Since these corrupted arguments are supposed to have only one correct answer, our protocol actually evaluates whether that answer could be retrieved. Aggregating all the test triples, we report the overall Hits@1, i.e., the proportion of cases in which the correct answer ranks first. The relation-specific Hits@1 is also reported, aggregated over the test triples with a specific relation.

The protocol is slightly different with that used in previous work [Bordes *et al.*, 2011; 2013]. First, instead of iterating through all the entities, we corrupt a test triple using only those entities with compatible types. Chang et al. [2014] have demonstrated that removing triples with incompatible types during test time leads to better results. Second, we choose to report Hits@1 rather than Hits@10 (i.e. the proportion of correct answers ranked in the top ten). Since in our case each corrupted argument has only one correct answer, we believe it is more substantive to evaluate whether that answer ranks first, as opposed to just in the top ten.

Implementation details. We implement RESCAL and TRESCAL in Java, and use the code released by Bordes et al. [2013] for TransE[5]. In RESCAL and TRESCAL, we fix the regularization parameter λ to 0.1, and the maximal number of iterations to 10, as suggested by Chang et al. [2014]. In TransE, we fix the margin to 1, the learning rate to 10, the batch number to 5, and the maximal number of iterations again to 10, which we found empirically to be enough to give the best performance. For each of the three models, we tune the the latent dimension d in the range of $\{10, 20, 30, 40, 50\}$ and select the optimal parameter setting.

We then incorporate rules into the three models with optimal parameter settings using ILP. To generate the objective, plausibility predicted by RESCAL or TRESCAL is normalized by $w'_{ij} = w_{ij}/\textsc{Max}$, and plausibility predicted by TransE is normalized by $w'_{ij} = (w_{ij} - \textsc{Avg}) / (\textsc{Max} - \textsc{Avg})$. The ILP problems on Location data finally get 21,540 variables and 18,192 constraints, and the ILP problems on Sport data 241,268 variables and 194,788 constraints. We use the lp_solve package[6] to solve the ILP problems. It takes about 1 minute on Location data and 2 hours on Sport data.

We repeat all experiments 10 times by drawing new training/test splits in each round. We report Hits@1 values averaged over the 10 rounds.

Quantitative results. Table 2 and Table 3 report the results on the test sets of the two data sets, where "H"/"T" indicates predicting head/tail entities for a relation. On Location data, the relation `StateHasCapital` gets a Hits@1 value of zero for all the methods, no matter predicting heads or tails. We remove that relation from Table 2. From the results, we can see that 1) The incorporation of rules significantly improves the performance of all the three embedding models on both data sets, with the Hits@1 values drastically enhanced. This observation demonstrates the superiority and generality of our approach. 2) RESCAL and TRESCAL perform better than TransE after incorporating rules, indicating that

[4] Since we only have a small number of triples, it is difficult to hold out a validation set with meaningful size.

[5] https://github.com/glorotxa/SME

[6] http://lpsolve.sourceforge.net/5.5/

Relation	RESCAL	r-RESCAL	TRESCAL	r-TRESCAL	TransE	r-TransE
CityCapitalOfCountry (H)	1.83	**92.91**	0.67	**83.17**	18.98	**59.38**
CityCapitalOfCountry (T)	6.26	**92.91**	1.67	**83.17**	28.48	**59.38**
CityLocatedInCountry (T)	9.11	**86.80**	6.53	**85.62**	25.30	**82.02**
CityLocatedInState (T)	**7.54**	0.00	**8.81**	1.43	**4.66**	1.67
StateLocatedInCountry (T)	57.43	57.43	56.88	56.88	3.64	3.64
Overall	14.11	**67.27**	12.71	**62.68**	16.55	**43.73**

Table 2: Overall and relation-specific Hits@1 (%) on Location data set.

Relation	RESCAL	r-RESCAL	TRESCAL	r-TRESCAL	TransE	r-TransE
AthleteLedSportsTeam (T)	42.03	**81.98**	42.16	**81.16**	10.43	**52.94**
AthletePlaysForTeam (T)	41.09	**78.88**	39.76	**78.31**	8.56	**54.50**
CoachesTeam (T)	2.53	**78.68**	2.62	**74.99**	14.43	**60.98**
OrganizationHiredPerson (H)	3.00	**68.65**	3.11	**69.78**	16.97	**51.43**
PersonBelongsToOrganization (T)	30.80	**72.82**	23.80	**73.57**	8.93	**54.23**
Overall	30.49	**78.17**	29.72	**77.26**	11.20	**54.58**

Table 3: Overall and relation-specific Hits@1 (%) on Sport data set.

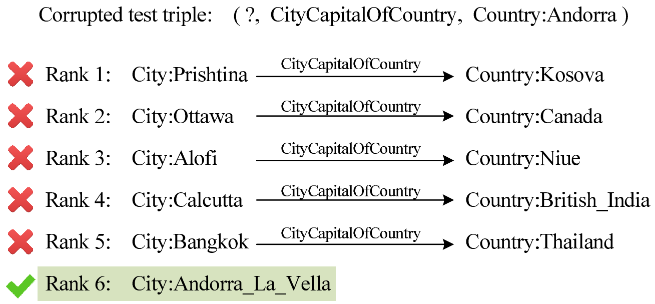
Corrupted test triple: (?, CityCapitalOfCountry, Country:Andorra)
✗ Rank 1: City:Prishtina CityCapitalOfCountry Country:Kosova
✗ Rank 2: City:Ottawa CityCapitalOfCountry Country:Canada
✗ Rank 3: City:Alofi CityCapitalOfCountry Country:Niue
✗ Rank 4: City:Calcutta CityCapitalOfCountry Country:British_India
✗ Rank 5: City:Bangkok CityCapitalOfCountry Country:Thailand
✓ Rank 6: City:Andorra_La_Vella

Figure 3: Case study on Location data set.

compared to margin-based ranking losses, embedding models based on reconstruction errors might be more preferred by the ILP framework. 3) The performance of TRESCAL is just comparable with that of RESCAL, showing that imposing the argument type expectation rule alone is not enough to bring better results. We need multiple rules, and our approach is definitely a proper choice for incorporating various rules.

Qualitative analysis. Figure 3 further provides a case study on the Location data set, to show how the rules can help an embedding model (i.e. RESCAL) in the entity retrieval task. Consider the test triple (City:Andorra_La_Vella, CityCapitalOfCountry, Country:Andorra) with the head corrupted. Without incorporating rules, RESCAL ranks the correct answer sixth, and the Hits@1 value associated with this case is zero. However, our training data tells that the answers ranked in the top five are capitals of other countries. By imposing the at-most-one restraint (i.e. a city can be the capital of at most one country), r-RESCAL rules out the top five answers and ranks the correct answer first. The Hits@1 value hence increases to one. This example indicates that rules are extremely useful in reducing the solution space during inference, and thus can greatly improve the performance of embedding models.

Consider this example again. Given the same test triple with the head corrupted, using the rules listed in Table 1 alone is not enough to retrieve the correct answer. Actually, based on the rules and the training data, only 75 among the total 94 cities which locate in or are capitals of other countries can be ruled out. The other 19 cities are all accepted by the rules. However, if we further consider the plausibility predicted by RESCAL, the correct answer will be promoted to the first. This example demonstrates the benefits of embedding models, resulted from learning and operating on latent variables. Our approach naturally preserves such benefits by integrating embedding models into the objective function.

4.3 Predicting New Facts

This task is to predict entire triples, rather than just heads or tails. We test RESCAL and TRESCAL in this task, before and after incorporating rules.[7]

On each data set, we use all the observed triples for model training, and ask human annotators to judge the correctness of the top N triples predicted by each of the methods. For RESCAL and TRESCAL, we rank all candidate triples (i.e. unobserved ones with compatible entity types) in descending order according to their plausibility, and return those ranked in the top N. For r-RESCAL and r-TRESCAL, we use an additional constraint $\sum_{k,i,j} x_{ij}^{(k)} \leq N'$ to conduct prediction. In all the methods, parameters are set to the optimal settings determined in Section 4.2. Note that in our data a same entity name can refer to different real-world entities (will be detailed in Section 4.4). Given a triple with an ambiguous entity name, we label it true as long as it holds for one real-world entity associated with the name. We report precision at the positions of 10, 20, 30, and 50.

Figure 4 reports the results on Location data. The results show that 1) r-RESCAL/r-TRESCAL consistently out-

[7] We have also tested TransE, but it does not perform well on this task, particularly before incorporating rules.

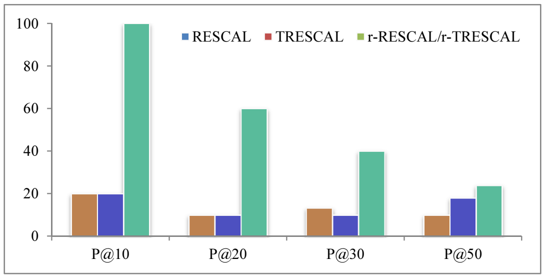

Figure 4: Precision (%) at position N on Location data set.

performs RESCAL/TRESCAL at all the positions. 2) The improvements are particularly significant for the topmost predictions. These observations again demonstrate the advantages of incorporating rules. We have conducted the same experiments on Sport data and observed similar phenomena.

4.4 Automatic Noise Detection

This task is to automatically detect noise in KBs. We test r-RESCAL and r-TRESCAL in this task on Location data, with the optimal parameter settings determined in Section 4.2. In both methods, all the observed triples are used for model training, and those getting $\epsilon = 1$ after solving the ILP problem are returned as noise. We then ask human annotators to judge the precision of the detection.

r-RESCAL identifies 22 suspicious triples, among which 21 are confirmed to be noise, with a detection precision higher than 95%. Table 4 lists the 21 triples, roughly categorized into three groups: factual errors, polysemy errors, and synonym errors. Factual errors refer to triples that are essentially false, e.g., the one stating that `City:Los_Angeles` locates in `State:Nevada`.[8] Polysemy errors are caused by the phenomenon that a same entity name can refer to different real-world entities. `City:Austin` is such an ambiguous entity. We observe two triples related to it, stating that it locates in `State:Colorado` and `State:Texas`. And as a matter of fact, there do exist two different cities named Austin, located in the two states respectively. Synonym errors are caused by the phenomenon that a same real-world entity may have different names. For instance, `State:Illinois` and `State:IL` refer to the same entity, with the latter being an abbreviation of the former. Ambiguous entities are marked by stars in Table 4. r-TRESCAL performs similarly to r-RESCAL, with 22 suspicious triples identified and 20 of them confirmed to be noise, getting a precision higher than 90%. The results demonstrate the effectiveness of the slack variables in detecting erroneous facts and ambiguous entities.

Head	Relation	Tail
City:Los_Angeles	CityLocatedInState	State:Nevada
City:Houston	CityLocatedInState	State:Utah
City:San_Antonio	CityLocatedInState	State:Utah
State:Pennsylvania	StateHasCapital	City:Philadelphia
State:Arkansas	StateHasCapital	City:Jonesboro
State:Florida	StateHasCapital	City:Miami
State:Florida	StateLocatedInCountry	Country:Tanzania
City:Austin ∗	CityLocatedInState	State:Colorado
City:Charleston ∗	CityLocatedInState	State:South_Carolina
City:Jackson ∗	CityLocatedInState	State:Mississippi
City:San_Marcos ∗	CityLocatedInState	State:California
State:Alabama	StateHasCapital	City:Montgomery ∗
State:Maryland	StateHasCapital	City:Annapolis ∗
State:Nebraska	StateHasCapital	City:Lincoln ∗
State:Nova_Scotia	StateHasCapital	City:Halifax ∗
State:Oregon	StateHasCapital	City:Salem ∗
State:Wyoming	StateHasCapital	City:Cheyenne ∗
City:Rockford	CityLocatedInState	State:IL ∗
City:Fort_Lauderdale	CityLocatedInState	State:FL ∗
City:Fresno	CityLocatedInState	State:CA ∗
City:Orlando	CityLocatedInState	State:FL ∗

Table 4: Noise detected on Location data set.

4.5 Discussions

Although powerful in the tasks described above, our approach gets two limitations. First, it can only cope with 1-To-Many, Many-To-1, or 1-To-1 relations, but cannot handle Many-To-Many relations. The reason is that without introducing the at-most-one constraints, the ILP problem given in Figure 2 tends to predict all triples with positive plausibility ($w_{ij}^{(k)} > 0$) to be true, leading to poor inference accuracy. Statistics show that in some KBs, only a small faction (lower than 25%) of the relations are Many-To-Many relations [Bordes *et al.*, 2013], making our approach still appealing for most cases.

Second, solving the ILP problem given in Figure 2 is time-consuming, which limits the efficiency and scalability of our approach. A possible solution is to decompose the ILP problem into multiple small-scale sub-problems for large KBs. Specifically, given that the objective function, the constraints R1, R2, and R3 are decomposable by different relations, we can simply decompose the ILP problem according to constraint R4. That is, two relations are assigned to a same group if one implicates the other. In this way, each of the resultant sub-problems corresponds to a small number of relations that implicate each other. Such a divide-and-conquer strategy is almost always practical for different KBs, and to some extent addresses the scalability issue.

5 Conclusion and Future Work

In this paper, we propose a novel KB completion approach which integrates embedding models and rules in a unified framework. It formulates inference as solving an ILP problem, where the objective function is generated from embedding models and the constraints are translated from rules. Facts inferred in this way are the most preferred by the embedding models, and at the same time comply with all the rules. The incorporation of rules significantly reduces the solution space and enhances the inference accuracy. To handle noise in KBs, a slacking technique is further proposed. We empirically evaluate our approach in entity retrieval and new fact prediction. Experimental results show significant and consistent improvements over state-of-the-art embedding models. Moreover, the slacking technique is demonstrated to be effective in automatic noise detection.

As future work, we plan to 1) Make our approach more efficient by acceleration or approximation, so as to handle larger KBs and more complicated rules such as $r_1(e_i, e_j) \wedge$

[8] Los Angeles is actually a city located in California.

$r_2(e_i, e_j) \Rightarrow r_3(e_i, e_j)$. 2) Investigate the possibility of incorporating rules *during* embedding rather than after embedding. It might result in more accurate embeddings, and benefit various tasks besides KB completion, e.g., relation extraction and entity resolution.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments and suggestions. This work is supported by the National Natural Science Foundation of China (grant No. 61402465), the Strategic Priority Research Program of the Chinese Academy of Sciences (grant No. XDA06030200), and the National Key Technology R&D Program (grant No. 2012BAH46B03).

References

[Bollacker *et al.*, 2008] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of SIGMOD*, pages 1247–1250, 2008.

[Bordes *et al.*, 2011] A. Bordes, J. Weston, R. Collobert, and Y. Bengio. Learning structured embeddings of knowledge bases. In *Proceedings of AAAI*, pages 301–306, 2011.

[Bordes *et al.*, 2013] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of NIPS*, pages 2787–2795, 2013.

[Bordes *et al.*, 2014] A. Bordes, X. Glorot, J. Weston, and Y. Bengio. A semantic matching energy function for learning with multi-relational data. *MACH LEARN*, 94(2):233–259, 2014.

[Bröcheler *et al.*, 2010] M. Bröcheler, L. Mihalkova, and L. Getoor. Probabilistic similarity logic. In *Proceedings of UAI*, pages 73–82, 2010.

[Carlson *et al.*, 2010] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr, and T. M. Mitchell. Toward an architecture for never-ending language learning. In *Proceedings of AAAI*, pages 1306–1313, 2010.

[Chang *et al.*, 2014] K. W. Chang, W. T. Yih, B. Yang, and C. Meek. Typed tensor decomposition of knowledge bases for relation extraction. In *Proceedings of EMNLP*, pages 1568–1579, 2014.

[Dittrich *et al.*, 2008] M. T. Dittrich, G. W. Klau, A. Rosenwald, T. Dandekar, and T. Müller. Identifying functional modules in protein-protein interaction networks: an integrated exact approach. *BIOINFORMATICS*, 24(13):223–231, 2008.

[Do *et al.*, 2007] M. B. Do, J. Benton, M. van den Briel, and S. Kambhampati. Planning with goal utility dependencies. In *Proceedings of IJCAI*, pages 1872–1878, 2007.

[Dong *et al.*, 2014] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of SIGKDD*, pages 601–610, 2014.

[Guo *et al.*, 2015] S. Guo, Q. Wang, L. Wang, B. Wang, and L. Guo. Semantically smooth knowledge graph embedding. In *Proceedings of ACL*, 2015.

[Jenatton *et al.*, 2012] R. Jenatton, N. L. Roux, A. Bordes, and G. R. Obozinski. A latent factor model for highly multi-relational data. In *Proceedings of NIPS*, pages 3167–3175, 2012.

[Jiang *et al.*, 2012] Shangpu Jiang, Daniel Lowd, and Dejing Dou. Learning to refine an automatically extracted knowledge base using markov logic. In *Proceedings of ICDM*, pages 912–917, 2012.

[Lao *et al.*, 2011] N. Lao, T. M. Mitchell, and W. W. Cohen. Random walk inference and learning in a large scale knowledge base. In *Proceedings of EMNLP*, pages 529–539, 2011.

[Lin *et al.*, 2015] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of AAAI*, pages 2181–2187, 2015.

[Miller, 1995] G. A. Miller. Wordnet: a lexical database for english. *COMMUN ACM*, 38(11):39–41, 1995.

[Nickel *et al.*, 2011] M. Nickel, V. Tresp, and H. P. Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of ICML*, pages 809–816, 2011.

[Pujara *et al.*, 2013] J. Pujara, H. Miao, L. Getoor, and W. Cohen. Knowledge graph identification. In *Proceedings of ISWC*, pages 542–557, 2013.

[Richardson and Domingos, 2006] M. Richardson and P. Domingos. Markov logic networks. *MACH LEARN*, 62(1-2):107–136, 2006.

[Riedel and Clarke, 2006] S. Riedel and J. Clarke. Incremental integer linear programming for non-projective dependency parsing. In *Proceedings of EMNLP*, pages 129–137, 2006.

[Rocktäschel *et al.*, 2014] T. Rocktäschel, M. Bošnjak, S. Singh, and S. Riedel. Low-dimensional embeddings of logic. In *Proceedings of ACL Workshop*, pages 45–49, 2014.

[Roth and Yih, 2004] D. Roth and W. T. Yih. A linear programming formulation for global inference in natural language tasks. In *Proceedings of CoNLL*, pages 1–8, 2004.

[Socher *et al.*, 2013] R. Socher, D. Chen, C. D. Manning, and A. Ng. Reasoning with neural tensor networks for knowledge base completion. In *Proceedings of NIPS*, pages 926–934, 2013.

[Suchanek *et al.*, 2007] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *Proceedings of WWW*, pages 697–706, 2007.

[Vossen *et al.*, 1999] T. Vossen, M. O. Ball, A. Lotem, and D. S. Nau. On the use of integer programming models in ai planning. In *Proceedings of IJCAI*, pages 304–309, 1999.

[Wang and Xu, 2013] Z. Wang and J. Xu. Predicting protein contact map using evolutionary and physical constraints by integer programming. *BIOINFORMATICS*, 29(13):266–273, 2013.

[Wang *et al.*, 2014] Z. Wang, J. Zhang, J. Feng, and Z. Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of AAAI*, pages 1112–1119, 2014.

[West *et al.*, 2014] R. West, E. Gabrilovich, K. Murphy, S. Sun, R. Gupta, and D. Lin. Knowledge base completion via search-based question answering. In *Proceedings of WWW*, pages 515–526, 2014.