A Data-Driven Approach to Infer Knowledge Base Representation for Natural Language Relations

Kangqi Luo, Xusheng Luo, Xianyang Chen, Kenny Q. Zhu*
Shanghai Jiao Tong University, Shanghai, China
{luokangqi,freefish_6174,st_tommy,kenzhu}@sjtu.edu.cn

Abstract

This paper studies the problem of discovering the structured knowledge representation of binary natural language relations. The representation, known as the schema, generalizes the traditional path of predicates to support more complex semantics. We present a search algorithm to generate schemas over a knowledge base, and propose a data-driven learning approach to discover the most suitable representations to one relation. Evaluation results show that inferred schemas are able to represent precise semantics, and can be used to enrich manually crafted knowledge bases. [1]

1 Introduction

Open Information Extraction (Open IE) is a recent popular technique that automatically mines relations between named entities from open-domain natural language data sources such as the world wide web. State-of-the-art Open IE systems, such as ReVerb [Fader *et al.*, 2011], NELL [Carlson *et al.*, 2010] and PATTY [Nakashole *et al.*, 2012], extract binary relations (e.g., "grandfather of", "was born in"), and has accumulated large ontologies of (e_{subj}, r, e_{obj}) *triple facts* called *relation instances*, where the subject and object arguments, e_{subj} and e_{obj}, are entity names and r is a lexico-syntactic pattern that connects the arguments in natural language and represents the relation. On the other hand, numerous community efforts have manually curated several comprehensive structured knowledge bases (KB) such as DBpedia [Auer *et al.*, 2007], Freebase [Bollacker *et al.*, 2008] and YAGO [Suchanek *et al.*, 2007], which are typically represented in the form of a graph, connecting unique named entities, concepts and their types using standard, predefined predicates as edges.

Though such knowledge bases are populated with millions of facts, they still face two key challenges. First, there are semantic gaps between KB predicates and natural language relations. For example, Freebase doesn't have a precise predicate for "has grandfather" relation, but instead has *parent* and *gender* predicates. Second, the knowledge base is far from being complete, which gives rise to an open research problem called knowledge base completion (KBC). The goal of KBC is to populate predicates in a KB with new facts, where both arguments of these new facts already exist in the knowledge base as entities. The answer to two challenges lies in the ability to learn a good knowledge base representation for a natural language relation, such as the one in Figure 1.

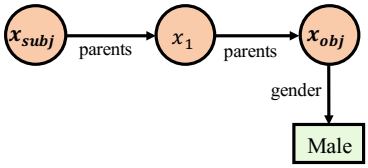


Figure 1: An example schema graph.

The state-of-the-art methods for this relation learning task can be categorized into two branches. The first branch is based on embedding technique, which learns a vector representation for every entity and predicate in the KB, then models relationships between two entities as vector translations in the embedding space. These methods are able to learn hidden semantics of a target relation. However, with large amounts of parameters, such embedding models require massive amount of training data and cause significantly more time for learning, which doesn't scale to a large KB.

The second branch is rule induction techniques, which use explicit rules as semantic representations of a target relation. Each rule is a substructure in the knowledge base, connecting the subject and object of the relation. One of the most intuitive structure is *path*: a sequence of predicates linking the subject to the object. One advantage of rule induction methods is that rules can be translated into SPARQL queries and hence all existing RDF tools can be applied. The other advantage is a relation with different subject and object types can be represented by multiple rules [Luo *et al.*, 2015]. Rules are also interpretable and human readable, which allows manual fine-tuning.

In this paper, we propose to generalize the path structure into a tree structure connecting not only the two target entities, but also other constants and constraints connected to the path. This tree structure is an abstraction of a set of all concrete sub-trees in the knowledge base having the same edge structure. We call such a tree structure a *schema graph*, or

[1] Kenny Q. Zhu is the contact author and is partially supported by NSFC grants 91646205 and 61373031.

schema in short. Figure 1 is an example schema graph, which is essentially a view on the knowledge base, joining several primitive predicates together.

Informally, the input of our task is a list of relation instances (e_{subj}, r, e_{obj}) extracted by natural language relation pattern r, and the output is a set of schema graphs with probabilities to measure their ability to represent the relation r in the knowledge base.

There are three technical challenges for inferring a schema representation. First, there are many possible schema graphs that potentially connect a pair of entities, and a brute force search over all schemas is intractable. Second, because the learning process is data driven, i.e., only relation instances and not schemas are available for training, the learning model needs to trade off between general, high recall but low precision schemas (e.g., parents + parents) versus specific, high precision but low recall schemas (e.g., parents + parents + gender=male). Third, with only positive training data which are entity pairs, the learning task is difficult. As a circumvention, one can use the closed world assumption to automatically generate negative instances. But due to data incompleteness of KB, potential false negatives may cause problem. This paper addresses these challenges and makes the following contributions.

- We define schemas as a generalized representation of natural language relations in knowledge base (Section 2);
- We present an effective local search based heuristic to generate a set of candidate schemas (Section 3.1);
- We propose a data-driven approach to model the schema inference problem as a querying task, and thus compute the probability distribution over schemas, given a natural language relation and its instances without explicitly generating negative training data (Section 3.2);
- The framework significantly outperforms previous best approaches on link prediction and triple classification task. The example results show that our schema inference model is able to discover concrete and precise semantics (Section 4).

2 Problem Definition

Definition 1. *A* Knowledge Base *is a triple $\langle E, L, P\rangle$, where: E is a finite set of all entities in KB; L is a finite set of all predicate names in KB; P is a finite set of predicate instances with the form $p(e_1, e_2)$ where $e_1, e_2 \in E$ and $p \in L$.*[2]

Definition 2. *A Schema (denoted by S), is a triple $\langle E', X, P_S\rangle$ where: $E' \subseteq E$, i.e., a subset of entities in KB respectively; X is a finite set of different variable entities, and each $x \in X$ indicates a placeholder for an entity $e \in E$; two special variables in X, x_{subj} and x_{obj}, denote the subject and object entity of the relation respectively; P_S is a finite set of schema predicate instances as $p_s(v_1, v_2)$ where $v_1 \in X$, $v_2 \in E' \cup X$ and $p_s \in L$. Moreover,*

[2]A special type of predicate in many knowledge bases is "IsA", which connects an entity with its *type*. However, for simplicity we treat a type as a special entity.

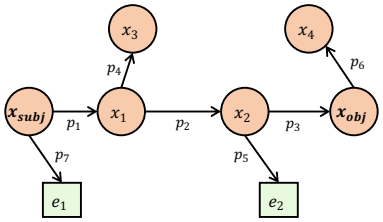


Figure 2: A general style of a schema graph.

- *S represents a directed tree structure, the root of which is the subject entity x_{subj};*
- *the path between x_{subj} to x_{obj} in the schema is called the* skeleton *of the schema;*
- *all edges other than the skeleton are called* constraints*;*
- *a schema with only a skeleton and no other edges, is called skeleton schema, or skeleton in short.*

Figure 2 shows a general style of a schema graph. Note that every edge of the graph must connect at least one variable entity. A *ground graph* is a subgraph of KB that is instantiated from a relation schema S, with every variable x_i instantiated with entity $e_i \in E$. If a ground graph of S instantiates x_{subj} to e_{subj} and x_{obj} to e_{obj}, then S is said to *cover* the entity pair (e_{subj}, e_{obj}).

Our problem is, given the KB, a natural language relation r along with a set of instances $\{(e_{subj}, r, e_{obj})\}$, induce a list of schemas and the probability distribution over the schemas, such that observed instances can be produced with the highest probability.

3 Approach

In this section, we discuss how a natural language relation is represented as knowledge base schemas. Given a relation r with a list of instances, we first generate a set of candidate schemas from these (e_{subj}, e_{obj}) pairs, and then discover the most suitable representations among them. Due to the lack of direct (relation, schema) training data, we propose a distant supervision approach and learn a probability distribution over all the candidates.

3.1 Candidate Schema Generation

We propose a search algorithm to collect candidate schemas from training relation instances. The intuition is that we first find suitable skeletons as a starting point, and then recursively add constraints on previous schemas, producing more specific candidate schemas.

We first use breadth-first search to find all suitable skeletons that connect each entity pair in KB. We limit the maximum length of the skeletons to be τ. The percentage of input relation instances covered by a schema S is called the *support ratio* of S, or $sup(S)$. To ensure the quality of the retrieved skeleton, we also require a schema to support at least γ percentage of training data, that is, $sup(S) \geq \gamma$. This limit helps the system filter out noisy schemas.

After candidate skeletons are produced, we deploy depth-first search on each skeleton to obtain more specific schemas. The challenge of schema expansion is that, the overall search

space is so large, even if the length of a skeleton is bounded. Inspired by beam search algorithm[Ney *et al.*, 1992], we introduce a priority queue to maintain the set of candidate schemas with high quality for each relation, and efficiently prune out unnecessary search spaces. Algorithm 1 shows the pseudo code of this step. Initially, Q is an empty priority queue, and always maintains the top-B schemas with largest coverages (line 7). The function $SchemaExpansion$ (line 8) takes S as input and returns a list of new schemas, each by adding one constraint to S.

Algorithm 1 Schema Searching

Input: Schema S, priority queue Q, budget B, minimum support ratio γ
Output: Priority queue Q after expanding on S

```
1: procedure SEARCH(S, Q, B, γ)
2:     if sup(S) < γ then
3:         return Q
4:     if Q.size < B or sup(S) > sup(Q.top) then
5:         Q.push(S)
6:         while Q.size > B do
7:             Q.pop()
8:         NewList ← SchemaExpansion(S)
9:         for S' in NewList do
10:            Q ← Search(S', Q, B, γ)
       return Q
```

Finally, we would like the generated set of candidate schemas to be diverse and not all similar to each other. We thus split the whole priority queue into isolated queues, one for each skeleton. The size of each queue is proportional to the support of the corresponding skeleton.

3.2 Schema Inference

For each relation r, after generating its candidate schemas, we now aim to learn the most suitable representation among them. It's a natural idea to compute the conditional probability over all the candidates. As a data-driven approach, we model the learning process as a query processing task: given the subject (or object) entity in an instance of r, use the schemas to query the best possible object (or subject) entity. In order to handle the trade-off between general and specific schemas, we use maximum likelihood estimation across all queries as the measurement to discover the most suitable schema distribution, which lead to the correct result and produce fewer irrelevant entities. The likelihood is defined as:

$$L(\vec{\theta}) = \prod_i P(o_i|s_i, \vec{\theta}) P(s_i|o_i, \vec{\theta}), \tag{1}$$

where $\vec{\theta}$ is the vector of schema probability distribution ($\sum_j \theta_j = 1$), whose length is the same as the number of candidate schemas of r, and s_i, o_i indicates the subject and object of the i-th entity pair, respectively.

We model $P(o|s, \vec{\theta})$ as a generative process: we first randomly choose a schema sc depending on Multinomial($\vec{\theta}$) (independent of s), then we query the schema on s, and o is randomly picked from the corresponding query results (independent of $\vec{\theta}$). With the conditional independences mentioned above, we define this generation step as:

$$\begin{aligned} P(o|s, \vec{\theta}) &= \sum_j P(sc_j|s, \vec{\theta}) P(o|s, sc_j, \vec{\theta}) \\ &= \sum_j \theta_j P(o|s, sc_j), \end{aligned} \tag{2}$$

where $P(o|s, sc_j)$ can be calculated directly from KB. Suppose $q(s, sc_j)$ is the query result set of j-th schema on the subject s, then o is uniformly selected from the set, with probability defined below:

$$P(o|s, sc_j) = \begin{cases} 1/\left|q(s, sc_j)\right| & o \in q(s, sc_j) \\ \alpha & \text{otherwise} \end{cases} \tag{3}$$

Here α is a smoothing parameter, since we don't want the likelihood to be 0. The similar formula holds for $P(s|o, \vec{\theta})$, which stands for the probability of querying a subject from object.

We have so far turned the schema inference problem into an optimization task: adjusting the probability distribution $\vec{\theta}$, such that likelihood function $L(\vec{\theta})$ is maximized. In order to solve the problem, we apply the RMSProp algorithm [Tieleman and Hinton, 2012] and iteratively search the best probability distribution. The algorithm converges after 500 iterations on average.

4 Experiments

In this section, we first evaluate the quality of our inferred schemas, then we perform experiments on the task of link prediction and triple classification. Finally, we discuss and analyze the errors in our system.

4.1 Experimental Setup

Knowledge bases. We use two knowledge bases throughout our experiments: **FB3m** and **FB15k**. FB15k [Bordes *et al.*, 2013] is a subset of Freebase containing 14,951 entities, 1345 predicates, and 483,142 triple facts. We use triples from training split as our knowledge base. Besides, we construct FB3m from the Freebase dump released of June 2015 [Google, 2015], which contains 3 million popular entities and 50 million triple facts (100 times larger than FB15k).
Relation datasets. We construct three relation datasets for experiments. The first two datasets, called "**PATTY-100**" and "**PATTY$^+$-100**", are extracted from PATTY [Nakashole *et al.*, 2012] OpenIE system, each containing 100 natural language relations. PATTY contains more than 200,000 different natural language relation synsets with millions of entity pairs extracted from Wikipedia. Each entity in PATTY is linked to Freebase through a unique Wikipedia page. PATTY-100 is for experiments related to FB15k, and contains relations with high support in PATTY such that all the entities can be found in FB15k. Conversely, PATTY$^+$-100 is for FB3m related experiments. It is sampled from all of PATTY and may contain more complex and long-tail relations. On average, each PATTY-100 relation has a support of 180 facts in FB15k, and each PATTY$^+$-100 relation has a support of 388 facts in FB3m. Both PATTY datasets are split into training / validation / testing sets (64% : 16% : 20%). The third dataset, called "**FB15k-37**", consists of 37 popular predicates in the

domains of "people", "location" and "sports", sampled from FB15k. FB15k-37 is a subset of FB122 [Guo *et al.*, 2016], removing predicates with too small testing data, and each predicate has at least 10 testing triples. Experiments on FB15k-37 treats our system as a classic KBC system.

State-of-the-art comparisons. For embedding techniques, we take TransE [Bordes *et al.*, 2013], KALE [Guo *et al.*, 2016], TEKE [Wang and Li, 2016] and HOLE [Nickel *et al.*, 2016] as our comparisons. For rule induction techniques, we compare with two models: SFE [Gardner and Mitchell, 2015] and AMIE+ [Galárraga *et al.*, 2015]. We also considered Coupled PRA model [Wang *et al.*, 2016] as a comparison. However, because different relations share almost no entity pairs in PATTY-100, the model would degenerate into the traditional PRA model and hence be strictly superseded by SFE.

Implementation details. We evaluate two variants of our approach: Ours-SC (producing schemas with constraints) and Ours-SK (schemas skeleton only) . For both variants, we set the maximum skeleton length $\tau = 3$, [3] the size of the priority queue to be 5000. We tune the minimum support γ in the range of {5%, 10%, 15%, 20%}, the smoothing parameter α in {1e-6, 1e-5, 1e-4} and the learning rate in {0.02, 0.05, 0.1} on the validation set. For comparison, we use the existing code for AMIE+ [4], and SFE system provided by Gardner et al. [2015] [5], KALE system by Guo et al. [2016], HOLE and TransE by Nickel et al. [2016] [6], and implement TEKE system by ourselves based on TransE. All the embedding systems adopt the max-margin model during the learning step. For KALE, we tune the learning rate in {0.02, 0.05, 0.1} and the margin in {0.1, 0.12, 0.15, 0.2}. For TransE, TEKE and HOLE, we tune the learning rate in {0.05, 0.1, 0.2} and the margin in {0.5, 1.0, 1.5, 2.0, 2.5}.

4.2 Schema Quality Evaluation

In this experiment, we focus on how the explicit semantic structure bridges the gap between Freebase and PATTY$^+$-100 relations. We compare the top schemas (a.k.a. rules) of 4 selected example relations produced by Ours-SC, Ours-SK, AMIE+ and SFE, all of which considered rule induction methods. The experiment is performed on FB3m, so that each model can find more different structures. For each relation from PATTY$^+$-100, we rank the candidate schemas from Ours-SC and Ours-SK by the learned probability distribution, while SFE ranks the path features by their feature weights, and AMIE+ ranks all the rules by their confidence score, which is the precision of the rule over triple facts in training data.

Figure 3 shows the comparison. We can learn from examples that: 1) The constraint edges help create more precise semantics. Compared with Ours-SK, the schema-based approach learns almost the perfect schemas on each example.

[3] We observed that $\tau > 3$ costs significantly more time with no substantial benefits.

[4] https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/amie/

[5] https://github.com/matt-gardner/pra

[6] https://github.com/mnick/scikit-kge

Table 1: AvgSc@n results on top-ranked schemas.

	n=1	n=3	n=5
Ours-SK	0.44	0.37	0.34
Ours-SC	**0.47**	**0.40**	**0.38**

2) The quality of top structures from AMIE+ and SFE is not as good as our results. AMIE+ ranks rules by confidence and hence prefers more specific rules. Once the search depth is increased to 4 or more, the system uses huge amount of memory and doesn't return. In SFE, the wildcard edge brings about a large amount of flexible path features, but most of them don't have clear semantics, and hard to construe by human.

As a complementary evaluation, we perform a human judge experiment on the quality of top ranked schemas produced by Ours-SC and Ours-SK. The evaluators are 3 non-author annotators who are familiar with Freebase. For each relation, up to top 5 schemas with a probability larger than 0.05 is labeled with a score in {0, 0.5, 1}, indicating "irrelevant schema" (semantic drift on the skeleton), "partial match" (the skeleton makes sense, but constraints can be improved) and "perfect match" (both skeleton and constraints are suitable), respectively. We first compute the average score for the top n schemas per relation and annotator, then average these scores over all relations and all annotators to produce AvgSc@n. The inter-annotator agreement is 0.541 by Kappa coefficient. As shown in Table 1, the schema-based approach improves the result by up to 13%.

4.3 Link Prediction

This task is to predict the missing object in the triple (e_{subj}, r, ?), or the missing subject in (?, r, e_{obj}). Following the evaluation protocol of KALE [Guo *et al.*, 2016], for each triple (e_{subj}, r, e_{obj}) in testing set, we replace e_{obj} by any other entities e'_{obj} in the knowledge base, forming a list of wrong triples (e_{subj}, r, e'_{obj}), with only one positive triple in it. By using Eq. (2), we calculate the score of each prediction in the list and rank them in descending order, returning the rank of the correct e_{obj} among all other wrong entities. Similarly, we can get the rank of e_{subj} at the subject side. To be consistent with the state-of-the-art systems, we aggregate over all testing triples, reporting the mean reciprocal rank (MRR) and the proportion of ranks no larger than n (Hits@n, or H@n for short). For each setting, we tune all the parameters based on the MRR score on the validation set.

Due to the existence of one-to-many relations, some wrong triples (e_{subj}, r, e'_{obj}) are actually correct and already observed in the dataset (either in training, validation or testing). In this case, we follow TransE [Bordes *et al.*, 2013] and create two settings called "raw" and "filtered". In the filtered setting, we remove such correct triples from the triple list before calculating the rank of each prediction. Conversely, in the raw setting, we don't remove any triples.

We perform link predication on FB15k in order to compare with embedding models. In the following experiments, we use $\gamma = 10\%$, $\alpha = 1e-4$ and learning rate = 0.1 as the best parameters to achieve the highest filtered MRR score on the validation set of PATTY-100. Table 2 and Table 3 shows the results for PATTY-100 and FB15k-37 respectively. SFE

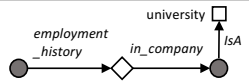
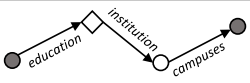
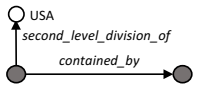
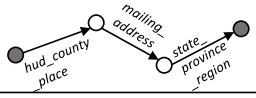
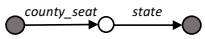
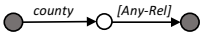
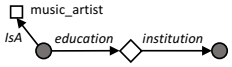
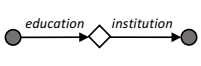
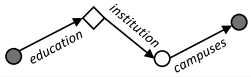
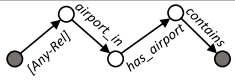
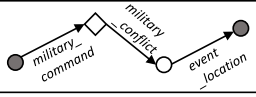
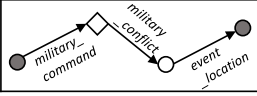
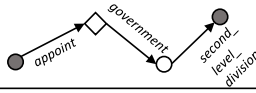
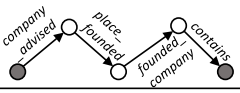
Natural Language Relation	Ours-SC	Ours-SK	AMIE+	SFE
(PER) taught at (LOC)				
(LOC) is a county located in (LOC)				
(PER) studied music at (LOC)				
(PER) 's invasion in (LOC)				

Figure 3: Top schemas produced by four systems on 4 complex relations. Circle node indicates entities or variables, the two black circles represents x_{subj} and x_{obj} respectively. Square node represents a type and diamond node represents a mediator (an n-ary predicate).

ran out of memory for both data sets because it needs to enumerate all possible entities in the KB, and hence is excluded from the tables. For PATTY-100 relations, our schema based approach outperforms both embedding and other rule induction models. Meanwhile, for FB15k-37 predicates, Ours-SK shows nearly the same performance as Ours-SC. That's because predicates in FB15k-37 may have an equivalent form in KB, for example, $location.location.containedby \rightarrow !location.location.contains$, where "!" indicates the reverse of a predicate, therefore skeletons are precise enough to represent such predicates. Furthermore, link predication results on natural language relations are relatively lower than those on KB predicates. Two possible reasons are: 1) Each predicate in FB15k-37 has up to thousands of supported instances in FB15k, while a natural language relation from PATTY-100 only has about 115 training instances. 2) Natural language relations are semantically more ambiguous than KB predicates, therefore triple facts of different semantics may be mixed during information extraction. On the other hand, knowledge bases are carefully curated and contain less ambiguity.

4.4 Triple Classification

This task is to predict whether a new triple (e_1, r, e_2) is correct or not. As a binary classification task, we need to generate negative triples for evaluations. Following the strategy used in KALE [Guo *et al.*, 2016], for each positive triple in the validation and test set, we generate 10 negative triples by randomly corrupting the entities, 5 at the subject position and 5 at the object position. We ensure that each corrupted entity has appeared in some other positive triples at the same position, and all corrupted triples do not exist in either the training, validation or testing set.

For each relation, we rank all positive and negative triples by their likelihood values (see Eq. (1)) in descending order, and calculate the average precision. We report the mean average precision (MAP) aggregated over all relations. We perform the experiment on FB15k, and Table 4 shows the result on different relation datasets.

Our system outperforms the other baseline methods on the PATTY-100 dataset, however, Ours-SK beats Ours-SC, because the negative triples cannot be distinguished by the constraints, hence the SC approach has no advantage. For example, instead of generating a child's own mother as a negative example, we generate the father of a random child.

Table 2: Link prediction results on PATTY-100 relations.

	Raw			Filtered		
	MRR	H@3	H@10	MRR	H@3	H@10
TransE	0.112	12.4	27.1	0.129	14.5	29.9
KALE	0.112	12.5	25.4	0.125	14.4	27.5
TEKE	0.101	10.9	24.1	0.114	12.6	26.3
HOLE	0.109	10.5	23.3	0.121	12.3	25.8
AMIE+	0.148	16.5	29.3	0.174	19.5	**31.9**
Ours-SK	0.169	18.2	29.3	0.179	19.1	30.4
Ours-SC	**0.172**	**18.5**	**29.8**	**0.185**	**19.9**	31.5

Table 3: Link prediction results on FB15k-37 relations.

	Raw			Filtered		
	MRR	H@3	H@10	MRR	H@3	H@10
TransE	0.310	39.3	53.2	0.394	52.5	65.0
KALE	0.342	40.6	53.0	0.410	48.7	60.6
TEKE	0.288	35.7	49.2	0.339	43.0	56.5
HOLE	0.234	26.7	39.5	0.323	36.5	50.5
AMIE+	0.395	46.1	53.7	0.562	60.0	68.9
Ours-SK	0.425	47.8	55.6	0.664	68.8	73.0
Ours-SC	**0.427**	**48.1**	**55.7**	**0.671**	**69.3**	**73.3**

Table 4: MAP results on triple classification task.

	PATTY-100	FB15k-37
TransE	0.304	0.666
KALE	0.309	0.654
TEKE	0.282	0.631
HOLE	0.308	0.680
SFE	0.329	0.621
AMIE+	0.226	0.730
Ours-SK	**0.408**	**0.804**
Ours-SC	0.403	0.803

4.5 Error Analysis

Our system may fail to capture the correct semantics in some natural language relations. We have analyzed the results in the above experiments and found several main causes of error.

1. Relation instances extracted by Open IE system could be incorrect. For example, given the relation *"served as"*, PATTY extracted an incorrect entity pair (William Dennison Jr., Ohio) from the sentence *"Dennison served as the 24th Governor of Ohio and as U.S. Postmaster General ..."*, while the correct object should be "Governor of Ohio".

2. PATTY relation synset mixed semantically different but lexically similar relation patterns, bringing ambiguity in the relation. For example, in PATTY's *"'s wife"* relation synset, we found a small list of instances where the object is actually husband, due to an opposite pattern *"the wife of"*. This phenomenon prevents us from finding correct gender constraints.

3. Knowledge base lacks necessary information for representing certain relations. Freebase doesn't hold knowledge about trivial relations like *"talk to"*, even for non-trivial relations, required predicates could be missing in Freebase. Given the relation *"(singer) performed in (LOC)"*, FB contains neither *place_visited* nor *hold_concerts_in* predicate, therefore it's hard to summarize into a precise representation.

4. Sometimes a meaningful schema is filtered out due to the search space limits in candidate searching step. In relation *"(actor) starring with (actor)"*, the length of the most suitable skeleton [7] is 4, hence the system failed to discover it under restriction $\tau = 3$.

5 Related Work

Knowledge base completion is an open research question since knowledge bases are far from being complete. These systems aim to complete the imperfectly extracted KB by predicting all entities e_2 which potentially have the target relation $rel(e_1, e_2)$ given the input e_1. By far most literature fall into two categories: *embedding* based and *rule* based.

Embedding based methods represent entities and relations in vector space and predict soundness of candidate triplets from these latent vectors. Among various embedding based models, there is a line of translation-based models such as TransE [Bordes *et al.*, 2013], TransH [Wang *et al.*, 2014], TransR [Lin *et al.*, 2015] and other enhanced but similar models. The general idea of these models is to train the embedding vectors under the criterion $\mathbf{h} + \mathbf{r} \simeq \mathbf{t}$. In order to take advantage of rich information in text corpus, TEKE [Wang and Li, 2016] can improve any translation-based method by adding an extra textual context embedding vector to the original representation, which is learned separately on the text corpus. HOLE [Nickel *et al.*, 2016] extends the system scalability by learning knowledge base within the framework of compositional vector space model.

Rule based methods try to directly make use of logic rules to infer relations. For example, $\text{parent}(x, y) \wedge \text{parent}(y, z) \rightarrow \text{grandparent}(x, z)$ is a generally acknowledged rule, and we can use this rule to find out more people with grandparent relations. Jiang et al. [2012] proposed a Markov logic based method for cleaning an automatically generated knowledge base. Pujara et al. [2013] proposed to use probabilistic soft logic (PSL) for this job. Galárraga et al. [2015] proposed AMIE+, a rule mining framework which directly mine Horn rules from KB triplets. Völker et al. [2011] proposed a statistical approach to induct schemas based on association rule mining. Other works treat rules as paths through entities in the KB. Lao et al. [2011] proposed Path Ranking Algorithm (PRA), which used a random walk path finding algorithm to map the target KB relation into a sequence of several basic relations. Subgraph feature extraction [2015], known as SFE, explores more than paths in the KB by exploring structural features around entities, and allows using a wildcard to indicate any possible edges. Wang et al. [2016] improved PRA with Coupled Path Ranking Algorithm (CPRA), where similar relations are clustered and jointly learned. However, in the experiment setting of this paper, relations in OpenIE dataset usually do not overlap and CPRA degenerates to PRA.

Some works in KBC combine above approaches by incorporating rules into embedding models. Logic rules are combined with embedding in KALE [Guo *et al.*, 2016], where the idea is to represent and model triples and rules in a unified framework. TRESCAL [Chang *et al.*, 2014] encodes type constraints into RESCAL [Nickel *et al.*, 2012]. Rocktäschel et al. [2015] proposed to embed first-order logic into low-dimensional vector spaces, and Wang et al. [2015] integrated KB embedding and rules with integer linear programming (ILP), with the objective function derived from the embedding model and constraints translated from rules.

In traditional KBC tasks, the target relation is an existing predicate in the KB, while we extend the definition of KBC into a broader scenario, since one may wish to add a new predicate (derived from natural language) into existing KB, and the Open IE system can help provide seed relation instance for further enrichment. In terms of mapping NL relation into KB, Zou et al. [2014] proposed an unsupervised TfIdf-based algorithm to figure out the mapping confidence of predicate paths to one relation. Zhang et al. [2012] also focused on learning path predicates using a Markov Logic Network [Richardson and Domingos, 2006]. While the above KBC systems focus on path representation, our work aims at understanding semantically complex relations and adopts complex schema with constraints.

6 Conclusion

This work mines the equivalence between natural language relations and structured knowledge known as schemas. It generalizes the simple path representation by adding constraints along the path and thus support more complex semantics. Experiments show that schema representation is able to describe the concrete and precise semantic meaning. Schemas thus learned have higher quality than those learned by existing rule induction approaches. Our approach can also be applied to the traditional knowledge base completion problem and yield good results.

[7] The skeleton is $actor \rightarrow med. \rightarrow film \rightarrow med. \rightarrow actor$.

References

[Auer *et al.*, 2007] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. *Dbpedia: A nucleus for a web of open data*. Springer, 2007.

[Bollacker *et al.*, 2008] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, pages 1247–1250, 2008.

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, pages 2787–2795, 2013.

[Carlson *et al.*, 2010] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, volume 5, page 3, 2010.

[Chang *et al.*, 2014] Kai-Wei Chang, Scott Wen-tau Yih, Bishan Yang, and Chris Meek. Typed tensor decomposition of knowledge bases for relation extraction. In *EMNLP*, pages 1568–1579, 2014.

[Fader *et al.*, 2011] Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *EMNLP*, pages 1535–1545, 2011.

[Galárraga *et al.*, 2015] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M Suchanek. Fast rule mining in ontological knowledge bases with AMIE+. *The VLDB Journal*, 24(6):707–730, 2015.

[Gardner and Mitchell, 2015] Matt Gardner and Tom Mitchell. Efficient and expressive knowledge base completion using subgraph feature extraction. In *EMNLP*, pages 1488–1498, 2015.

[Google, 2015] Google. Freebase data dumps. https://developers.google.com/freebase/data, 2015.

[Guo *et al.*, 2016] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. Jointly embedding knowledge graphs and logical rules. In *EMNLP*, pages 1488–1498, 2016.

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

[Lao *et al.*, 2011] Ni Lao, Tom Mitchell, and William W Cohen. Random walk inference and learning in a large scale knowledge base. In *EMNLP*, pages 529–539, 2011.

[Lin *et al.*, 2015] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187, 2015.

[Luo *et al.*, 2015] Kangqi Luo, Xusheng Luo, and Kenny Q. Zhu. Inferring binary relation schemas for open information extraction. In *EMNLP*, pages 555–560, 2015.

[Nakashole *et al.*, 2012] Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. PATTY: a taxonomy of relational patterns with semantic types. In *EMNLP-CoNLL*, pages 1135–1145, 2012.

[Ney *et al.*, 1992] Hermann Ney, Reinhold Haeb-Umbach, B-H Tran, and Martin Oerder. Improvements in beam search for 10000-word continuous speech recognition. In *ICASSP*, volume 1, pages 9–12, 1992.

[Nickel *et al.*, 2012] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing yago: scalable machine learning for linked data. In *WWW*, pages 271–280, 2012.

[Nickel *et al.*, 2016] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. Holographic embeddings of knowledge graphs. In *AAAI*, 2016.

[Pujara *et al.*, 2013] Jay Pujara, Hui Miao, Lise Getoor, and William Cohen. Large-scale knowledge graph identification using psl. In *AAAI Fall Symposium on Semantics for Big Data*, 2013.

[Richardson and Domingos, 2006] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine learning*, 62(1-2):107–136, 2006.

[Rocktäschel *et al.*, 2015] Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. Injecting logical background knowledge into embeddings for relation extraction. In *HLT-NAACL*, pages 1119–1129, 2015.

[Suchanek *et al.*, 2007] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *WWW*, pages 697–706. ACM, 2007.

[Tieleman and Hinton, 2012] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2), 2012.

[Völker and Niepert, 2011] Johanna Völker and Mathias Niepert. Statistical schema induction. In *Extended Semantic Web Conference*, pages 124–138. Springer, 2011.

[Wang and Li, 2016] Zhigang Wang and Juanzi Li. Text-enhanced representation learning for knowledge graph. In *AAAI*, pages 1293–1299, 2016.

[Wang *et al.*, 2014] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119, 2014.

[Wang *et al.*, 2015] Quan Wang, Bin Wang, and Li Guo. Knowledge base completion using embeddings and rules. In *IJCAI*, pages 1859–1866, 2015.

[Wang *et al.*, 2016] Quan Wang, Jing Liu, Yuanfei Luo, Bin Wang, and C Lin. Knowledge base completion via coupled path ranking. In *ACL*, pages 1308–1318, 2016.

[Zhang *et al.*, 2012] Congle Zhang, Raphael Hoffmann, and Daniel S Weld. Ontological smoothing for relation extraction with minimal supervision. In *AAAI*, 2012.

[Zou *et al.*, 2014] Lei Zou, Ruizhe Huang, Haixun Wang, Jeffer Xu Yu, Wenqiang He, and Dongyan Zhao. Natural language question answering over rdf: a graph data driven approach. In *SIGMOD*, pages 313–324. ACM, 2014.