Fine-grained Type Inference in Knowledge Graphs via Probabilistic and Tensor Factorization Methods

A B M Moniruzzaman
Queensland University of Technology
Brisbane, Australia
a.moniruzzaman@qut.edu.au

Richi Nayak
Queensland University of Technology
Brisbane, Australia
r.nayak@qut.edu.au

Maolin Tang
Queensland University of Technology
Brisbane, Australia
m.tang@qut.edu.au

Thirunavukarasu Balasubramaniam
Queensland University of Technology
Brisbane, Australia
thirunavukarasu.balas@qut.edu.au

ABSTRACT

Knowledge Graphs (KGs) have been proven to be incredibly useful for enriching semantic Web search results and allowing queries with a well-defined result set. In recent years much attention has been given to the task of inferring missing facts based on existing facts in a KG. Approaches have also been proposed for inferring types of entities, however these are successful in common types such as 'Person', 'Movie', or 'Actor'. There is still a large gap, however, in the inference of fine-grained types which are highly important for exploring specific lists and collections within web search. Generally there are also relatively fewer observed instances of fine-grained types present to train in KGs, and this poses challenges for the development of effective approaches. In order to address the issue, this paper proposes a new approach to the fine-grained type inference problem. This new approach is explicitly modeled for leveraging domain knowledge and utilizing additional data outside KG, that improves performance in fine-grained type inference. Further improvements in efficiency are achieved by extending the model to probabilistic inference based on entity similarity and typed class classification. We conduct extensive experiments on type triple classification and entity prediction tasks on Freebase FB15K benchmark dataset. The experiment results show that the proposed model outperforms the state-of-the-art approaches for type inference in KG, and achieves high performance results in many-to-one relation in predicting tail for KG completion task.

KEYWORDS

Knowledge Graph, Type Inference, Fine-grained Type, Semantic Web Search, Tensor Factorization

ACM Reference Format:
A B M Moniruzzaman, Richi Nayak, Maolin Tang, and Thirunavukarasu Balasubramaniam. 2019. Fine-grained Type Inference in Knowledge Graphs via Probabilistic and Tensor Factorization Methods. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313597

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313597

1 INTRODUCTION

Knowledge Graphs (KGs) mainly describe real-world objects and their interrelations in various topical domains by organized in graphs. KGs are web-scale probabilistic knowledge bases that store billions of facts about people, places, and things that are often represented in the form of a triplet (head entity, relation, tail entity). The way KGs are enriching information about the real worlds objects and connections by combine extractions from web contents (obtained via analysis of text, tabular data, page structure, and human annotations) with prior knowledge derived from existing knowledge repositories [42] [47]. Examples of prominent large-scale KGs are Yago [54] [28] [11], Freebase [16] [3], DBpedia [14] [37] [1], NELL [20] [5], Google's Knowledge Graph, Microsoft's Satori, Probase [7] , and Google Knowledge Vault [27]. These KGs have become powerful assets for improving Web search by providing richer data for Entity Pane, Carousel, and Facts Across Segments in the search panel. Additionally, KGs are becoming important resources for different Artificial Intelligence (AI) and Natural Language Processing (NLP) applications. Despite their seemingly large size, these knowledge graphs are still incomplete with missing information, such as type, properties, and relations [42] [47] [23] [57] [63] [50].

Types in KG are used to express the concept of classes. According to KG idiomatic usage, a KG object "has X, Y, Z types" is equivalent to an object "is a member of the X, Y, Z classes". In the case of U.S. President Barack Obama, the KG object for Obama would have the types *person* and *US president* to indicate that the object is a member of the Persons and U.S. Presidents. Fine-grained types are sub-classes of type classes in KG. For example, a type class *'Actor'* may has *American Actor, Film Actor, TV Actor, Hollywood Actor, Female Actor, Voice Actor* or *Comedy Actor* sub classes. The Actor typed entities in KG are therefore can be further defined with these fine-grained types.

Defining fine-grained types of entities in KG allows Web search queries with a well-defined result set. When someone doing a search on the web, the search engines need to understand the words what they refer to real-world things, and by this, search engines can do a better search with just the contents when someone looks up on the web. The types of entities in KG provide the meaning of

the words, such as building is a building object and animal are animal objects; they are not just a random string of characters. Fine-grained typed entities and entities with complete set of types in KG, make possible to explore lists and collections with web searching by 'Knowledge Graph Carousel' in popular search engines. For example, you can explore, how many woman won the Noble prize? or list of Renaissance painters.

Type inference is concerned with inferring the existence (or probability of existence) of missing types for instances (objects or entities) in KGs. In a KG, not every entity may come with proper type information. Untyped entities and entities with incomplete set of types are a common problem in Semantic Web KGs [42] [47]. In recent years much attention has been observed in KG completion task that automatically infer missing facts based on existing facts in a KG [23] [57]. Approaches have also been proposed for inferring types of entities, however these are successful in common types such as 'Person', 'Movie', or 'Actor'. There is still a large gap, however, in the inference of fine-grained types which are highly important for exploring specific lists and collections within web search. Generally, very limited numbers of observed fine-grained types data (RDF-triples with fine-grained type information) present in KGs, and this poses a high challenge in developing approaches for inferring entities of the fine-grained types.

This paper proposes a new approach based on a tensor model for fine-grained type inference. This new approach is explicitly modeled for leveraging domain knowledge and utilizing additional data outside KG, that allows a ranked representation for type inference and result in richer data for tensor factorization. Further improvements in efficiency are achieved by extending the model to *probabilistic inference* based on entity similarity and typed class classification. We evaluate our approach with two standard evaluation tasks *type triple classification* and *entity prediction* on benchmark dataset Freebase FB15K dataset [4] and FB15K-237 Knowledge Base Completion Dataset [2]. The experiment results show the proposed model outperforms the state-of-the-art approaches for type inference in KG (10% to 19% on different fine-grained types compare to Rescal [43]), and achieve high results in many-to-one relation in predicting tail for KG completion task.

2 RELATED WORK

Given the importance of missing type assertions task, there are some approaches based on *Reasoning*, *Probabilistic Method* [49], *Hierarchical Classification Approach* [40] [41] [60], *Association Rule Mining* [32] [46], *Topic Modelling* [52], *Support Vector Machine (SVM)* [51], *k-nearest neighbors classifier (KNN)* [45], and *Tensor Factorization* [44] have been proposed and applied on different in KGs are summarized in Table 1. All approaches have merits and demerits in performance for missing type assertions task in KG.

In [52],the use of *Topic Modeling* for type prediction is proposed. Entities in a KG are represented as documents, on which *Latent Dirichlet Allocation (LDA)* [15] is applied for finding topics. In [45], [51] authors exploit interlinks between the KGs, and Wikipedia predict types in a KG. In [46], exploit such association rules to predict missing types in DBpedia based on redundancies different type systems of KGs. As topics, the interlinks and the type redundancies between KGs can only be covered limited associated top level types, these approaches are not capable for fine-grained type inference.

Some approaches have been proposed for type inference in KG such as SDType [49] a heuristic approach, SNCN [41], a hierarchical classification approach. Since SDType is based on statistical method and there are not enough facts exits in KG for fine-grained types [48], [49], this approach is not capable to produce meaningful results in predicting fine-grained types. Fine-grained types basically are not systematically structured with a type hierarchy in KGs and SNCN relay on the hierarchical type structure in KG, this approach is therefore not efficient in fine-grained type inference. Besides, some KGs (such as Freebase) are not systematically structured with a type hierarchy, in this case, this approach is not useful for this task. Though, SDType, and SNCN algorithm is successful in predicting higher-level classes (Actor), while predicting fine-grained classes (such as VoiceActor or AmericanActor) is much more difficult. None of these methods are able to capture the interactions of latent features that obtain probabilistic likelihoods of type-relations existing between entities (objects) in KG. Tensor modeling, a well-known approach to represent and analyze latent relationships inherent in multi-dimensional data [35].

RESCAL [43] [44] is the state-of-the-art method for link prediction and type inference in KGs that has been used for type inference on YAGO [11] entire KG [44]. Though, RESCAL has been achieved good performance among the state-of-the-art approaches for type inference in KG, the performance of type inference (fine-grained types) are significantly lower than the performance of top level (common) types in KG. Furthermore, this method and other existing methods for type inference in KG are not able to utilize the domain knowledge in KG and the additional sources of information outside of KG. However, for type inference in KG, it has not been explored that the utilization of additional sources of information by combining knowledge base and text. Our approach is based on tensor factorization method which is specially designed for fine-grained type inference in KG. This approach is able to capture interactions of entities outside of KG and to learn type-domain knowledge in KG. Furthermore, this model is designed to learn type hierarchy structure and similarities of entities for type; and utilize these information to improve fine-grained type inference results in KG.

3 UTILIZING ADDITIONAL DATA

Semantic Web KGs are represented in the form of the Resource Description Framework (RDF) [8] triples <*subject, predicate, object*>, where the subject and object are entities and the predicate is the relation type, (e.g. <*Steve Jobs, founderOf, Apple*>). Uniform Resource Identifier (URI) uniquely identifying each entity in Semantic Web KGs. For instance, entity *Donald Trump* can be found as *http://dbpedia.org/page/Donald_Trump* in DBpedia, and as *https://en.wikipedia.org/wiki/Donald_Trump* in Wikipedia. Every entity in Semantic Web KGs (e.g. DBpedia or Freebase) has anchor link to Wikipedia page, where a document of information is available for an associated entity.

Each entity in Freebase is uniquely identified by a MID (Machine ID). For instance, */m/0cqt90* as MID is used for *Donald Trump* entity. Using MID and the predicate *http://www.w3.org/2002/07/owl#sameAs*

for each entity in Freebase, we can issue a SPARQL query to retrieve the corresponding DBpedia, or Wikipedia URIs. For example, Donald Trump, the current President of USA, is mapped via the triple by SPARQL query. The texts (information) available for an associated entity can be represented as full lexicalized dependency paths. For instance, the entity in KG such as *Donald Trump*, or *United States* has textual information in Wikipedia page, or other online news repositories (e.g. New York Times or ClueWeb), as illustrated in Fig 1.

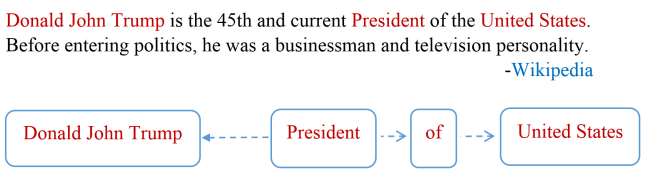

Figure 1: A full lexicalized dependency path from textual data.

Figure 1 shows the lexicalized dependency path between two entities that occur together in a sentence. An instance of textual relation of type "<nsubj> president <prep> of <obj> United States", corresponding to the sentence is added as additional data based on this occurrence. These full lexicalized dependency path from textual data are used as relations between Freebase entities by mapping with MID. For instance, the entity */m/0cqt90* (Donald Trump) in RDF triple relationship in Freebase as: (*/m/0cqt90, <President_of>, /m/01kw0b*). In addition to KG some other relations can be obtained from other sources by mapping MID. Relations connected to entity */m/0cqt90* (MID) from additional data sources (see in Fig 2). Relations come from additional data sources (outside from KG) may be useful information about entities in KG. From modelling perspective, tensor representations are appealing to RDF KGs because they provide an elegant way to represent multiple RDF triples.

We propose the *weighted-tensor* interpretation scheme to effectively model KG and text-based data for constructing the tensor. Without applying weight in tensor construction, the objective function would place equal importance on fitting both observed and unobserved values. As KGs are typically very large scale, and constructed tensors are therefore often very sparse, this will result in fitting a large number of unobserved data and uncertainty in the observations. The weighting tensor prevents this from happening by emphasizing only the observed (or certain) entries. Each RDF triples in KG expresses facts (complete meaningful information) about real world object, and whereas each text based triples provides less important facts (auxiliary information) about the real world object. The observed data for factorization is therefore obtained from combination of KG and other dataset which contains textual relations. The interpretation of RDF KG can be interpreted as a tensor, where first mode of a tensor therefore models the occurrences of all entities as a subject, the second mode models the occurrences of all entities as an object, and the third mode models the different relations. The incorporation of additional data for type inference in KG becomes possible via a tensor factorization model where KG and additional data consisting of e entities and r different relations can then be represented in form of a tensor $X \in R^{e \times e \times r}$ with entities

$$X_{i,j,k} = \begin{cases} 2, & \text{if the relationship } r_k(e_i; e_j) \text{ exists in KG.} \\ 1, & \text{if the relationship } r_k(e_i; e_j) \text{ exists in additional data.} \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

The values ('2' and '1') and ('0') of X_{ijk} come from tensor model are regarded as observed and un-observed data respectively for tensor factorization. Weighting the tensor by this scheme allows a ranked representation for type inference and result in richer data for tensor factorization, that leads to an improvement in efficiency for fine-grained type inference.

4 LEVERAGING DOMAIN KNOWLEDGE

Most KGs store facts about real-world objects covering only numbers of specific domains (e.g. *"Movie"*, *"Book"*, or *"Place"*). For instance, types in KG such as *actor, film, director,* or *producer* and fine-grained types such as *filmActor, TVActor, regularActor, guestActor, executiveDirector, AssistantProducer* are in *"film"* domain. Given the importance the fine-grained inference task in KG, typed entities (objects) for given fine-grained types in one domain (such as *"film"* domain) are less likely to be entities in other domains (such as *"book"*, or *"place"*). For instance, inferring entities for fine-grained types (such as regularActor, guestActor) would be a typed in Actor, those entities generally are in same domain in KG.

The collaborative activities between the entities in *"film"* domain are therefore higher importance for fine-grained type inference in this domain. However, interactions among entities with different relation types can be among different domains as well, as types of entities in KG may be come from different domains. Domain information can be obtained from KG, such as in Freebase each relation start with domain e.g. */film/film/starring/film/performance/actor*, and */tv/tv_program/regular_cast/tv/regular_tv_appearance/actor*; in DBPedia domain information can be found in the property of a relation by SPARQL [9] query with *http://www.w3.org/1999/02/22-rdf-syntax-ns#Property*. For example, a set of actor typed entities can be a set of author typed where these entities are involved with different domains. Ignoring these information (inter-domains collaboration activities of entities) may effect on latent features learning by factorization. Modelling domain knowledge (the interactions of entities in the domain, and inter-domins) is therefore prevents this happening and leverages the domain information in factorization for an improvement in fine-grained type inference.

We propose the *domain-relevance weighted tensor (DrWT)* method for fine-grained type inference, that explicitly model a mapping from the domain-based representation to the observed tensor in factorization. Given the KG and additional data along with list of labelled domain, *DrTW* classifies the entries correspondence to the domains in KG and additional dataset.

DrWT Algorithm:

Input: *Knowledge Graph* $G = (D, E, R)$*; where set* D *is a set of domains in KG,* $D = \{d_1, ..., d_{|d|}\}$*; set* E *is a set of entities (objects),* $E = \{e_1, ..., e_{|e|}\}$*, and* E_s *as subject set of entities which occur as subject in relation links, where* $E_s \in E$*; and and* E_o *as object set of entities which occur as object in relation links where* $\{E_o \in E\}$ *and, the set* R *is a set of relations (Predicates) between entities,* $R = \{r_1, ..., r_{|r|}\}$.

Output: set ∇T_d is a list of triples $<< (E_s)_i, (R)_i, (E_o)_i >>$ for each $d_i \in D$ (Domains)
Start
01: **function** DomainRelevance (D, E, R)
02: $d \leftarrow domain$
03: **for each** $r_i \in R$ **do**
04: **for each** $r_i \in d$ **do**
05: $source\ (e_s) : R \rightarrow E_s$ ▷ return source(e)
06: $target\ (e_o) : R \rightarrow E_o$ ▷ return target(e)
07: $e_d \leftarrow (e_s \cup e_o)$
08: **while** $e_i \in e_d$ OR $r_i \in d$ **do**
09: $T_d \leftarrow$ SELECT-TRIPLES $\{(e_s)_i, (r)_i, (e_o)_i\}$
10: **end while**
11: $\nabla T_d \leftarrow T_d$
12: **end for**
13: **end for**
14: **end function**

Applying *DrWT* the populated tensor is then represented the collaborative activities of the type domain can be modelled subject, object, predicate and domain in 4th-order tensor presentation. Once the 4th-order tensor representations are constructed applying DrWTF method, the next steps is latent factors generation via tensor factorization, where tensor factorization technique is applied to learn effectively the latent factors for fine-grained type inference.

5 FINE-GRAINED TYPE INFERENCE WITH TENSOR MODEL

We use CP [21] [30] based *4th-order Tensor Factorization* for deriving the latent relationships between dimensions of the tensor model. After latent factors generation via tensor factorization, we therefore follow *tensor reconstruction* process to reveal new entries that are inferred from the latent factors. the *4th-order Tensor Factorization* model, CP factorizes each forth-order domain-relevance weighted tensor $\mathcal{X} \in \mathbb{R}^{S \times O \times P \times D}$, can be formulated as

$$f(\mathbf{E}, \mathbf{F}, \mathbf{G}, \mathbf{H}) = \|\mathcal{X} - \mathbf{E}, \mathbf{F}, \mathbf{G}, \mathbf{H}\|^2 \quad (2)$$

where $\mathcal{X}$ is factorized using a CP model on each mode-n matricization and results in four latent factors matrices, $\mathbf{E} \in \mathbb{R}^{S \times R}$, $\mathbf{F} \in \mathbb{R}^{O \times R}$, $\mathbf{G} \in \mathbb{R}^{P \times R}$ and $\mathbf{H} \in \mathbb{R}^{D \times R}$ corresponding to the each dimensions of tensor $\mathcal{X}$. R is the column size of the corresponding latent factor matrices; S, O, P and D are denoted as *Subject, Object, Predicate* and *Domain* of entries to tensor. The diagonal core tensor $\hat{\mathcal{C}} \in \mathbb{R}^{R \times R \times R \times R}$, that defines the interaction between the *Subject, Object, Predicate* and *Domain* can be calculated by multiplying all the latent factors. Afterwords, the reconstructed tensor $\hat{\mathcal{X}}$ is derived multiplying the core tensor by all latent factor matrices:

$$\hat{\mathcal{X}} := \hat{\mathcal{C}} \times_s \hat{\mathbf{E}} \times_p \hat{\mathbf{F}} \times_o \hat{\mathbf{G}} \times_d \hat{\mathbf{H}} \quad (3)$$

Where the core tensor $\hat{\mathcal{C}}$ and the feature matrices $\hat{\mathbf{E}}$, $\hat{\mathbf{F}}$, $\hat{\mathbf{G}}$ and $\hat{\mathbf{H}}$ are the model parameters that have to be learned and $\mathcal{X}$ is the tensor product to multiply a matrix on dimension x with a tensor.

The task of fine-grained type inference in KG is to infer which subject entities (s) of a type relation (p) is missing to object entity (o). That means that the model has to infer from the values of $\hat{x}_{s,p,o,d}$ of the domain tensor $\hat{\mathcal{X}}$ (from equation no. 5). Instead of inferring single element (subject entities) from $\hat{\mathcal{X}}$, this model provides a list of best possible (ranked) elements (subject entities) of a fine-grained type relation. Given a reconstructed tensor $\hat{\mathcal{X}}$ the list of top of the N highest scoring entities for a given fine-grained type relation (p) and an object entity (o)

$$Top(N, p, o, d) := argmax^N_{t \in T}\{\hat{x}_{s,p,o,d}\} \quad (4)$$

To infer entities for fine-grained type inference, we define appropriate thresholds for each of fine-grained type relations. These thresholds are therefore used as metric in the inference model so that the missing entries can be regarded as either "relevant" (score higher than the threshold) or "irrelevant" (score lower than the threshold) entries from the revealed new entries that are inferred from the latent factors. In order to further improve the performance of inference to the missing entities for fine-grained types, the inference model is extended to *probabilistic inference* approach with multilabel classification of type classes.

5.1 Probabilistic Inference with Collective Multilevel Type Classification and Typed Entity Similarity

The probabilistic inference in an alternative approach to define the probability of one set typed entities to be other fine-grained typed by observing data from classification of type classes, and the latent similarity of entities in KG. In this case, the list of fine-grained types is ranked and generated based on the probability score of an entity to select candidate entities, in which the entity typed classes, and entity similarity to the fine-grained type classes are taken into account. The underlying idea of probabilistic ranking is that collective multilabel classification of entities where entities are allowed to have more than one class types and related entities often share identical classes in RDF KG. For instance, en entity of *Actor* typed is also typed of *Artist, Person,* and *Thing* classes (see in Fig. 2). Fine-grained type inference further can be cast as a multilabel

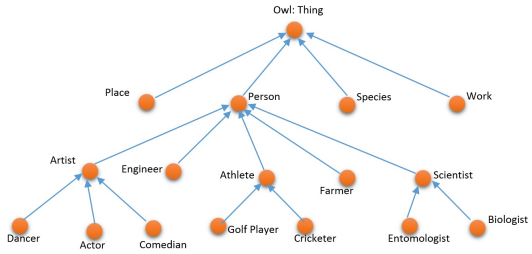

Figure 2: A part of DBPedia type hierarchy.

classification problem, by introducing type classes as entities to the data, and inferring the probability of relationships *rdf-type*.

As RDF KG is multi-relational data, the similarity of entities is therefore determined by the similarity of their relationships, following the intuition that "if two objects are in the same relation to the same object, this is evidence that they may be the same object" The collaborative activities of entities as subjects $\mathbf{A_s} \in \mathbb{R}^{S \times P}$ and objects $\mathbf{A_o} \in \mathbb{R}^{O \times P}$ in relations in a domain can be modelled by the entity matrix $\hat{\mathbf{A}}$, where $\hat{\mathbf{A}}$, is QR matrix factorization [22] [29] of $\sum(A_s + A_o)$. For each domain the the latent space $\hat{\mathbf{A}}$ therefore

reflects the similarity of entities in the relational domain. The type or fine-grained type classes set $C_e = \{t_1, t_2, t_3, ...t_n\}$ where C_e is a set of Types in one KG. A list of type or fine-grained type classes that are considered for given fine-grained type. For each fine-grained type in C_e the candidate entities set, $\hat{E}_t = \{\hat{e}_1, \hat{e}_2, \hat{e}_3, ...\hat{e}_n\}$ where E_t is a set of typed entities in one KG.

We use the Bayes' theorem [26] [61] for predicting the class candidate entity E_t that have the highest posterior probability given C_e, $p(C_e|E_t)$. The posterior probability is utilized to calculate the preference probability of an entity e to be fine-grained typed t in C_e type classes by observing current type classes of entity e, and latent similarity of entity e to fine-grained typed entity. The conditional probability can be formulated as:

$$p(C_e|E_t) = \frac{p(E_t|C_e)p(C_e)}{p(E_t)} \tag{5}$$

where *prior probability* $p(C_e)$ is the prior distributions of parameter set C_e in a single domain before E_t is observed, that is relative frequency with which observations from that class occur in a population. Generally, prior probability for fine-grained type classes are lower compared to top level type classes in KG. $p(C_e|E_t)$ is the joint probability of observing type class preference set C_e given E_t, and entity similarity preference given fine-grained type t. Using the assumption of multinomial event model distribution for the Naive Bayes classifier, the posterior probability p_{e_n,t_r} for fine-grained type t_e with fine-grained type class C_e for candidate entity e_n, an instance of E_t, is obtained by multiplying the *prior probability* of t_e, $P(C_e = t_e)$, with the probability of preference candidate entity e_n, an instance of E_t, given t_e, $P(e_n|C_e = t_e)$:

$$p_{e_n,t_e} = p(t_e|C_e) = \sum_{t=1}^{|C_e|} P(e_n|C_e = t_e) \prod_{\hat{e}=1}^{|\hat{E}_e|} P(\hat{E}_e|C_e) \tag{6}$$

where, $P(\hat{E}_e|C_e)$ is probability of likelihood for t_e in C_e, is derived from the entities set, $\hat{E}_t = \{\hat{e}_1, \hat{e}_2, \hat{e}_3, ...\hat{e}_n\}$ where values from reconstructed tensor $\hat{X}$, and entity similarity values from $\hat{A}$ are used.

Algorithm: Probabilistic Inference

Input: *Initial tensor* $\mathbf{X} \in \mathbb{R}^{S \times O \times P \times D}$, *reconstructed tensor* $\hat{X}$, *the type or fine-grained type classes set* $C_e = \{t_1, t_2, t_3, ...t_n\}$, *and number N rank list for top candidate entities.*

Output: *A list of top entities set* $\nabla\hat{E}_t$ *for each fine-grained type* t_i.

Start:

01: Get the candidate entities set $\hat{E}_t = \{\hat{e}_1, \hat{e}_2, \hat{e}_3, ...\hat{e}_n\}$:

$\hat{E}_t \leftarrow \{\hat{X} \,|(t_i \in \hat{X}$ AND $t_i \in C_e)$ AND $(\hat{X} \leftarrow \hat{x}_{s,p,o,d})\}$

02: Get the similarity of entities set $\hat{A}_{e(t)} = \{\hat{a}_1, \hat{a}_2, \hat{a}_3, ...\hat{a}_n\}$:

$\hat{A}_{e(t)} \leftarrow \{\hat{A} \,|(e(t)_i \in \hat{A}$ AND $e(t)_i \in \hat{X})\}$

03: ***function*** CalculatePosteriorProbability $(\hat{X}, \hat{E}_t, \hat{A}_{e(t)}, N)$

04: ***while*** $e_i \in \hat{E}_t$ ***do***

05: $p_{e_n,t_e} \leftarrow p(t_e|C_e)$ ▷ *from Equation no.* (9) *where,* $P(\hat{E}_e|C_e)$ *is probability of likelihood for* t_e *in* C_e

06: ***end while***

07: ***for*** $n \leftarrow 1toN$ ***do***

08: $\hat{E}_t \leftarrow argmax^N_{e_i \in \hat{E}_t} \{p_{e_n,t_e}\}$

09: ***if*** $n \leftarrow 1toN$ ***do***

10: ***end for***

11: $\nabla\hat{E}_t \leftarrow \hat{E}_t$

14: ***end function***

Further accuracy improvements are therefore achieved by extending the inference model by probabilistic inference with collective multilevel type classification and typed entity similarity.

6 EMPIRICAL EVALUATION

We evaluate our approach on Freebase FB15K dataset [4] and FB15K-237 Knowledge Base Completion Dataset [2] (see Table 1). Freebase [16] [3] is a large collaborative knowledge base consists of a large number of the world facts, such as triplets (*Donald Trump, PresidentOf, United States*) and (*Johnny Depp, profession, actor*). The FB15K (Bordes et al. 2013), is a subset of Freebase which has been commonly used to evaluate various KG completion models [18] [58] [39] [39] [33] [38] [31] [59]. In the FB15K-237 Knowledge Base Completion Dataset, the triples (entity- textual-entity) are derived from 200 million sentences from the ClueWeb12 corpus coupled with Freebase entity. There are around 3.9 million text descriptions corresponding to the relation types in Freebase. The FB15K-237 dataset has been used in [56] [55] [62] for embedding representations for textual relations with Freebase entity mention annotations.

Table 1: Datesets used in the experiments.

Datasets	# Entities	# Relations	# Triples
FB15K	14,951	1,345	486,641
FB15K-327	14,951	2,766,477	3,977,677

For implementation, we use *tensor-toolbox* [10] and *poblano-toolbox* [6] in Matlab. We apply *weighted tensor scheme* in constructing tensor where the tensor size (14951 x 14951 x 2,767,822) with 486541 entries from KG (FB15K); and 4460819 entries from textual dataset (FB15K-237). We then apply *domain-relevance weighted tensor (DrWT)* to construct 4th order tensor with domain entries, where the tensor size (14951 x 14951 x 2,767,822 x 52) . We use CP [21] [30] based *4th-order Tensor Factorization* for the latent factor generation, and use CP-ALS algorithm [24] [34] [25] for computing tensor factorization. Since domain information is not depended in one other dimension of the tensor, we use 4th order tensor factorization instead of using *Coupled Matrix Tensor Factorization (CMTF)* [12] [13]. We also apply *non-negativity constrain* [36] for effectively interpreting factor components from tensor factorization.

Two standard tasks, *Entity Prediction* and *Type Triple Classification* as are proposed to evaluate our approach for fine-grained type inference. The triple classification task e.g. type triple classification for fine-grained type which was first introduced by by Socher et al in [53]. The link prediction task which is standard task for KG completion evaluation e.g entity inference for fine-grained type, was first introduced by Bordes et al. in [18]. Since then these standard tasks have been used to evaluate various KG completion models, such as Rescal [43], TransE [18], TransH [58], TransR [39], CTransR [39], TransD [33], PTransE [38], and KG2E [31].

6.1 Type Triple Classification Task

We evaluate our approach with *Type Triple Classification* aims to judge whether a given triplet (h, r, t) with fine-grained type relation

is correct or not, which is a binary classification task. In FB15K, types of a person entity can be obtained from a triples by both (/people/person/profession) relation and by (/people/profession/people_with_his_profession) relation. Fig. 10 shows some different type triplets distributions with both relations in *Film, Television,* and *Music* domains in FB15K.

We evaluate our approach by type triple classification task on FB15K with six general types and eight fine-grained types in three different domains: *'Film'*, *'TV'* and *'Music'* with the relevant state-of-the-art method RESCAL [43] [44]. As we have mentioned previous section, RESCAL has been achieved good performance among the state-of-the-art approaches for type inference in KG [44]. As SDType [49] and SNCN [41] are not latent factor model, we can not directly compare our model with them.

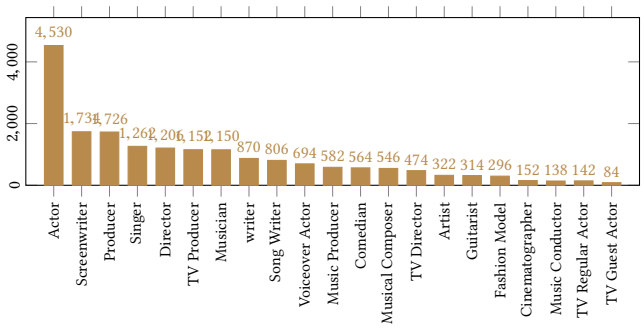

Figure 3: Different Type Triplets Distributions in FB15K.

Evaluation protocol. We follow the same protocol introduced by Socher et al in NTN [53]. Evaluation of classification needs negative labels. As FB15k has not released negative triples in previous works, we construct negative triples by corrupting the corresponding positive (observed) triples. This is followed with the same procedure used in [53]. Dissimilarity between triple and typed triple classifications is that, we only consider to corrupt observed triples that convey types or fine-grained types of entities as shown in Fig. 3. Corrupting observed triples for all relations is not required experiment for evaluation to our approach, as it is out of the scope of this work. The decision rule for classification is simple: for each type triple (h; r; t), if the similarity score (obtained by the models) is below a relation-specific threshold T, then the triple will be classified as positive. Otherwise, it will be classified as negative.

Discussion. From Table 2, we can see that our approach outperforms RESCAL significantly in all the reported top level types and fine-grained types. Some observations are as follows:

(a) From top class types, we can see that both RESCAL and our approach have achieved very high performance results for inferring *actor* type on FB15K. This is because high number of observed triples with actor type object (see Fig. 10), and therefore associated other relationships to actor type object present in FB15K, that positively impact both models for correctly scoring positive and negative triples.

(b) However, from fine-grained class types, we can see that our approach improves results for *Film Actor* and *TV Actor* significantly (11%) and (16%) respectively. Since *Film Actor* and *TV Actor* are in different domains (Film and TV), these domain entries for corresponding triples to the tensor model is defined by DrWT method. This prevents bias scoring positive and negative triples for *Film Actor* to *TV Actor* and vice-versa. Similar effect by DrWT method on results for *TV Director* and *TV Producer* fine-grained type inference.

(c) From fine-grained class types, we can see that our approach our approach significantly outperforms Rescal for *Regular TV Actor* and *Guest TV Actor* where very less number of observed triples with these fine-grained types object present in FB15K (see Fig. 10).

Table 2: Types Inference Comparison with RESCAL

Top Class Types		
	RESCAL	**Our Approach**
Actor	98.0	99.0
Director	90.0	96.0
Producer	89.0	93.0
Musician	78.0	89.0
Singer	74.0	81.0
Writer	70.0	76.0
Fine-grained Types		
	RESCAL	**Our Approach**
TV Director	82.0	91.0
Voiceover Actor	60.0	79.0
Song Writer	79.0	88.0
TV Producer	81.0	90.0
Film Actor	87.0	98.0
TV Actor	77.0	93.0
Regular TV Actor	58.0	78.0
Guest TV Actor	49.0	68.0

6.2 Entity Prediction Task

Entity prediction aims to predict the missing head (h) or tail (t) entity given relation type (r) and other entity, i.e. predicting h given (?; r; t) or predicting t given (h; r; ?) where ? denotes the missing element. In this task, for each position of missing entity, the system emphasizes more on ranking a set of candidate entities from the KG, instead of only giving one best result. In testing phase, for each test triple (h; r; t), we replace the head/tail entity by all entities in the KG, and rank these entities in descending order of similarity scores calculated by the models.

In FB15K, relations can be categorized according to the cardinalities of their head and tail arguments into four classes: one-to-one, one-to-many, many-to-one, and many-to-many. A given relationship is one-to-one if a head can appear with at most one tail, one-to-many if a head can appear with many tails, many-to-one if many heads can appear with the same tail, or many-to-many if multiple heads can appear with multiple tails (see in Table 3).
Evaluation protocol. We follow the same protocol in TransE [18]: For each testing triplet (h; r; t), we replace the tail t by every entity e (14951 in FB15K) in the KG and calculate a similarity score obtained by the model on the corrupted triple (h; r; e). Ranking the scores in ascending order, we then get the rank of the original correct triplet. Aggregated over all the testing triplets, the proportion of

Table 3: Examples of four types of relations in FB15K

Types of Relation	Relations in FB15K
one-to-one	/location/country/capital
one-to-many	/people/profession/people_with_his_profession /film/production_company/films /music/genre/artists /film/director/film
many-to-one	/people/person/place_of_birth /film/film/production_companies /people/person/profession /film/film/directed_by
many-to-many	/film/film/starring/film/performance/actor /film/actor/film/film/performance/film /film/actor/film

ranks not larger than 10 (denoted as Hits@10) metric is reported for evaluation. For this metric a higher Hits@10 is better.

Since the dataset is same, we directly report the experimental results of several baselines from the literature, (see Table no. 4). Instead of taking overall link prediction task results, we take link prediction reported results for predicting heads and prediction tails based on all types of relations: *one-to-one (1-1), one-to-many (1-N), many-to-one (N-1)* and *many-to-many (N-N)*) separately.

Table 4: Evaluation results of Predicting Head (HITS@10) by mapping properties of relations on FB15K

Dataset	**FB15K**			
Task	**Predicting Head**			
Metric	**(HITS@10)**			
Relation Category	**1-1**	**1-N**	**N-1**	**N-N**
Unstructured [17]	34.5	2.5	6.1	6.6
SE [19]	35.6	62.6	17.2	37.5
SME [17]	30.7	69.6	19.9	38.6
TransE [18]	43.7	65.7	18.2	47.2
TransH [58]	66.8	87.6	28.7	64.5
TransR [39]	78.8	89.2	34.1	69.2
CTransR [39]	81.5	89.0	34.7	71.2
PTransE [38]	90.1	92.0	58.7	**86.1**
TransD [33]	86.1	**95.5**	39.8	78.5
KG2E [31]	**92.3**	93.7	**66.0**	69.6
Our Approach	68.0	94.0	30.0	65.0
Task	**Predicting Tail**			
Metric	**(HITS@10)**			
Relation Category	**1-1**	**1-N**	**N-1**	**N-N**
Unstructured [17]	34.3	4.2	1.9	6.6
SE [19]	34.9	14.6	68.3	41.3
SME [17]	28.2	13.1	76.0	41.8
TransE [18]	43.7	19.7	66.7	50.0
TransH [58]	65.5	39.8	83.3	67.2
TransR [39]	79.2	37.4	90.4	72.1
CTransR [39]	80.8	38.6	90.1	73.8
PTransE [38]	90.1	**70.7**	87.5	**88.7**
TransD [33]	85.4	50.6	94.4	81.2
KG2E [31]	**92.6**	67.9	94.4	73.4
Our Approach	50.0	38.6	**96.0**	68.5

Discussion. For the comparison of Hits@10 of different kinds of relations, Table 4 shows the detailed results (entity prediction for heads and tails in triples) by mapping properties of relations on sparse dataset, FB15k. In predicting tail, we can see that our approach outperforms baseline approaches, e.g. TransE, TransH, TransR, CTransR significantly in *many-to-one relation*. In predicting head, we can see that our approach produces good results in *one-to-many relation*. KG2E outperforms other approaches in one-to-one relations, and PTransE has achieved highest results in one-to-many and many-to-many relations in predicting tail on FB15K. Compared with other types of relations, our approach does not outperform previous methods for one-to-one, one-to-many and many-to-many relations in predicting tail. This is because, our approach explicitly considers many-to-one relation for predicting tail, as this type of relationship between entities can only convey types or fine-grained types of entities in KG. For example, in the KGs, such as Freebase, DBPedia types of entities e.g. *Johnny Depp* can form in a KG triple as (Johnny Depp <subject entity> , type <relation> , actor <object entity>). In Freebase in DBPedia, type relation expresses as <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>. Fine-grained type inference is therefore ideologically comparable by many-to-one relation for predicting tail in KG.

7 CONCLUSION

This paper proposes a new approach based on a tensor model for fine-grained type inference in KG. This approach is explicitly modelled for leveraging domain knowledge in KG, and for utilizing the additional large amount of interactions among multiple entities and text descriptions. We further develop probabilistic inference based on collective multilevel type classification and latent similarity of typed entities. Experiment results show that utilizing additional data outside of KG, leveraging domain knowledge with tensor modelling; and developing 'probabilistic inference' by observing data from classification of type classes, and the latent similarity of entities significantly improves performance for inferring fine-grained types of entities in KG. Results in type triple classification and entity prediction tasks show the proposed model outperforms the state-of-the-art approaches for type inference in KG (10% to 19% on different fine-grained types compare to the state-of-the-art approach Rescal), and achieves high results in many-to-one relation in predicting tail for KG completion task.

REFERENCES

[1] "DBPedia" Public Semantic Knowledge Graph. http://wiki.dbpedia.org/. Accessed: August 08, 2017.
[2] "FB15K-237 Knowledge Base Completion Dataset. https://www.microsoft.com/en-us/download/details.aspx?id=52312. Accessed: September 29, 2017.
[3] "Freebase" downloadable version of the data in Freebase. https://developers.google.com/freebase/. Accessed: August 06, 2017.
[4] "Freebase" Knowledge Graph. https://developers.google.com/freebase/. Accessed: September 29, 2017.
[5] "NELL" Never-Ending Language Learning Project. http://rtw.ml.cmu.edu/rtw/index.php. Accessed: August 05, 2017.
[6] "Poblano Toolbox" Poblano - Sandia software - Sandia National Laboratories. https://software.sandia.gov/trac/poblano/. Accessed: August 29, 2018.
[7] "Probase" Knowledge Base. https://www.microsoft.com/en-us/research/project/probase/. Accessed: September 29, 2018.
[8] "RDF" Resource Description Framework. https://www.w3.org/RDF/. Accessed: September 29, 2018.
[9] "SPRQL" Query Language for RDF. https://www.w3.org/TR/rdf-sparql-query/. Accessed: September 29, 2018.
[10] "Tensor Toolbox" MATLAB Tensor Toolbox. https://www.sandia.gov/~tgkolda/TensorToolbox/index-2.6.html. Accessed: August 09, 2018.

[11] "YAGO" semantic knowledge base. http://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/. Accessed: August 09, 2017.

[12] Evrim Acar, Tamara G Kolda, and Daniel M Dunlavy. All-at-once optimization for coupled matrix and tensor factorizations. *arXiv preprint arXiv:1105.3422*, 2011.

[13] Evrim Acar, Morten Arendt Rasmussen, Francesco Savorani, Tormod Næs, and Rasmus Bro. Understanding data fusion within the framework of coupled matrix and tensor factorizations. *Chemometrics and Intelligent Laboratory Systems*, 129:53–63, 2013.

[14] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. *The semantic web*, pages 722–735, 2007.

[15] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[16] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. AcM, 2008.

[17] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259, 2014.

[18] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795, 2013.

[19] Antoine Bordes, Jason Weston, Ronan Collobert, Yoshua Bengio, et al. Learning structured embeddings of knowledge bases. In *AAAI*, volume 6, page 6, 2011.

[20] 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.

[21] J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of âĂIJeckart-youngâĂİ decomposition. *Psychometrika*, 35(3):283–319, 1970.

[22] Tony F Chan. Rank revealing qr factorizations. *Linear algebra and its applications*, 88:67–82, 1987.

[23] Liang Chang, Manli Zhu, Tianlong Gu, Chenzhong Bin, Junyan Qian, and Ji Zhang. Knowledge graph embedding by dynamic translation. *IEEE Access*, 5:20898–20907, 2017.

[24] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-ichi Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.

[25] Lieven De Lathauwer and Dimitri Nion. Decompositions of a higher-order tensor in block termsâĂŤpart iii: Alternating least squares algorithms. *SIAM journal on Matrix Analysis and Applications*, 30(3):1067–1083, 2008.

[26] Pedro Domingos and Michael Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine learning*, 29(2-3):103–130, 1997.

[27] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–610. ACM, 2014.

[28] MS Fabian, K Gjergji, WEIKUM Gerhard, et al. Yago: A core of semantic knowledge unifying wordnet and wikipedia. In *16th International World Wide Web Conference, WWW*, pages 697–706, 2007.

[29] Ming Gu and Stanley C Eisenstat. Efficient algorithms for computing a strong rank-revealing qr factorization. *SIAM Journal on Scientific Computing*, 17(4):848–869, 1996.

[30] Richard A Harshman. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970.

[31] Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. Learning to represent knowledge graphs with gaussian embedding. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 623–632. ACM, 2015.

[32] Jochen Hipp, Ulrich Güntzer, and Gholamreza Nakhaeizadeh. Algorithms for association rule miningâĂŤa general survey and comparison. *ACM sigkdd explorations newsletter*, 2(1):58–64, 2000.

[33] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 687–696, 2015.

[34] Hyunsoo Kim, Haesun Park, and Lars Eldén. Non-negative tensor factorization based on alternating large-scale non-negativity-constrained least squares. In *Bioinformatics and Bioengineering, 2007. BIBE 2007. Proceedings of the 7th IEEE International Conference on*, pages 1147–1151. IEEE, 2007.

[35] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[36] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788, 1999.

[37] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195, 2015.

[38] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. Modeling relation paths for representation learning of knowledge bases. *arXiv preprint arXiv:1506.00379*, 2015.

[39] 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.

[40] André Melo, Heiko Paulheim, and Johanna Völker. Type prediction in rdf knowledge bases using hierarchical multilabel classification. In *Proceedings of the 6th International Conference on Web Intelligence, Mining and Semantics*, page 14. ACM, 2016.

[41] Andre Melo, Johanna Völker, and Heiko Paulheim. Type prediction in noisy rdf knowledge bases using hierarchical multilabel classification with graph and latent features. *International Journal on Artificial Intelligence Tools*, 26(02):1760011, 2017.

[42] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.

[43] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 809–816, 2011.

[44] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing yago: scalable machine learning for linked data. In *Proceedings of the 21st international conference on World Wide Web*, pages 271–280. ACM, 2012.

[45] Andrea Giovanni Nuzzolese, Aldo Gangemi, Valentina Presutti, and Paolo Ciancarini. Type inference through the analysis of wikipedia links. In *LDOW*, 2012.

[46] Heiko Paulheim. Browsing linked open data with auto complete. 2012.

[47] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, 8(3):489–508, 2017.

[48] Heiko Paulheim and Christian Bizer. Type inference on noisy rdf data. In *International Semantic Web Conference (1)*, volume 8218, pages 510–525, 2013.

[49] Heiko Paulheim and Christian Bizer. Improving the quality of linked data using statistical distributions. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 10(2):63–86, 2014.

[50] Riccardo Porrini, Matteo Palmonari, and Isabel F Cruz. Facet annotation using reference knowledge bases. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 1215–1224. International World Wide Web Conferences Steering Committee, 2018.

[51] Jennifer Sleeman and Tim Finin. Type prediction for efficient coreference resolution in heterogeneous semantic graphs. In *Semantic Computing (ICSC), 2013 IEEE Seventh International Conference on*, pages 78–85. IEEE, 2013.

[52] Jennifer Sleeman, Tim Finin, and Anupam Joshi. Topic modeling for rdf graphs. In *LD4IE@ ISWC*, pages 48–62, 2015.

[53] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934, 2013.

[54] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM, 2007.

[55] Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, 2015.

[56] Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, 2015.

[57] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, 2017.

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

[59] Han Xiao, Minlie Huang, and Xiaoyan Zhu. Transg: A generative model for knowledge graph embedding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2316–2325, 2016.

[60] Ruobing Xie, Zhiyuan Liu, and Maosong Sun. Representation learning of knowledge graphs with hierarchical types. In *IJCAI*, pages 2965–2971, 2016.

[61] Harry Zhang. The optimality of naive bayes. *AA*, 1(2):3, 2004.

[62] Jingyuan Zhang, Chun-Ta Lu, Bokai Cao, Yi Chang, and S Yu Philip. Connecting emerging relationships from news via tensor factorization. In *Big Data (Big Data), 2017 IEEE International Conference on*, pages 1223–1232. IEEE, 2017.

[63] Kaja Zupanc and Jesse Davis. Estimating rule quality for knowledge base completion with the relationship between coverage assumption. In *Proceedings of the Web Conference 2018*, pages 1–9, 2018.