This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

Knowledge Graph Embedding: A Survey of Approaches and Applications

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo

Abstract—Knowledge graph (KG) embedding is to embed components of a KG including entities and relations into continuous vector spaces, so as to simplify the manipulation while preserving the inherent structure of the KG. It can benefit a variety of downstream tasks such as KG completion and relation extraction, and hence has quickly gained massive attention. In this article, we provide a systematic review of existing techniques, including not only the state-of-the-arts but also those with latest trends. Particularly, we make the review based on the type of information used in the embedding task. Techniques that conduct embedding using only facts observed in the KG are first introduced. We describe the overall framework, specific model design, typical training procedures, as well as pros and cons of such techniques. After that, we discuss techniques that further incorporate additional information besides facts. We focus specifically on the use of entity types, relation paths, textual descriptions, and logical rules. Finally, we briefly introduce how KG embedding can be applied to and benefit a wide variety of downstream tasks such as KG completion, relation extraction, question answering, and so forth.

Index Terms—Statistical relational learning, knowledge graph embedding, latent factor models, tensor/matrix factorization models.

1 Introduction

Recent years have witnessed rapid growth in knowledge graph (KG) construction and application. A large number of KGs, such as Freebase [1], DBpedia [2], YAGO [3], and NELL [4], have been created and successfully applied to many real-world applications, from semantic parsing [5], [6] and named entity disambiguation [7], [8], to information extraction [9], [10] and question answering [11], [12]. A KG is a multi-relational graph composed of entities (nodes) and relations (different types of edges). Each edge is represented as a triple of the form (*head entity*, *relation*, *tail entity*), also called a *fact*, indicating that two entities are connected by a specific relation, e.g., (`AlfredHitchcock`, `DirectorOf`, `Psycho`). Although effective in representing structured data, the underlying symbolic nature of such triples usually makes KGs hard to manipulate.

To tackle this issue, a new research direction known as *knowledge graph embedding* has been proposed and quickly gained massive attention [13], [14], [15], [16], [17], [18], [19]. The key idea is to embed components of a KG including entities and relations into continuous vector spaces, so as to simplify the manipulation while preserving the inherent structure of the KG. Those entity and relation embeddings can further be used to benefit all kinds of tasks, such as KG completion [14], [15], relation extraction [20], [21], entity classification [13], [22], and entity resolution [13], [18].

Most of the currently available techniques perform the embedding task solely on the basis of observed facts. Given a KG, such a technique first represents entities and relations in a continuous vector space, and defines a scoring function on each fact to measure its plausibility. Entity and relation embeddings can then be obtained by maximizing the total plausibility of observed facts. During this whole procedure, the learned embeddings are only required to be compatible within each individual fact, and hence might not be predictive enough for downstream tasks [23], [24]. As a result, more and more researchers have started to further leverage other types of information, e.g., entity types [25], [26], relation paths [27], [28], [29], textual descriptions [30], [31], [32], [33], and even logical rules [23], [34], [35], to learn more predictive embeddings.

In this article, we provide a thorough review of currently available KG embedding techniques, including those that use facts alone, as well as those that further leverage additional information. We further introduce how the learned embeddings can be applied to and benefit a wide variety of entity-oriented tasks. Nickel et al. [36] have made a survey of statistical relational learning methods on KGs, including embedding techniques, path ranking algorithms [37], [38], [39], and Markov logic networks [40], [41], [42]. In contrast to their work, we focus specifically on KG embedding, and make a systematic review of existing techniques, including not only the state-of-the-arts but also those with latest trends. Particularly, we make the review based on the type of information used in these techniques.

The rest of this article is organized as follows. Section 2 briefly introduces basic notations. Section 3 reviews techniques that conduct embedding using only facts observed in KGs. We describe the overall framework, specific model design, typical training procedures, as well as pros and cons of such techniques. Section 4 discusses embedding techniques that further incorporate other information besides facts. We focus on the use of entity types, relation paths, textual descriptions, and logical rules. Section 5 further explores the application of KG embedding in downstream tasks like KG completion, relation extraction and question answering. Finally, we present our concluding remarks in Section 6.

- *Q. Wang, Z. Mao, B. Wang, L. Guo are with the Institute of Information Engineering, Chinese Academy of Sciences (CAS) and the School of Cyber Security, University of CAS. Q. Wang is also with the State Key Laboratory of Information Security, CAS.*
  E-mail: {wangquan,maozhendong,wangbin,guoli}@iie.ac.cn

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

2 NOTATIONS

Throughout this article, we use a boldface lower-case letter $\mathbf{x}$ to represent a vector, with its i-th entry denoted as $[\mathbf{x}]_i$. The ℓ_p norm of a vector for $p \geq 1$ is denoted as $\|\mathbf{x}\|_p$, and $\|\mathbf{x}\|_{1/2}$ means either the ℓ_1 norm or the ℓ_2 norm. Let $\text{diag}(\mathbf{x})$ be a diagonal matrix, the i-th diagonal entry of which is $[\mathbf{x}]_i$. Let $|\mathbf{x}|$ denote the absolute value function, $\tanh(\mathbf{x})$ the hyperbolic tangent function, and $\text{ReLU}(\mathbf{x})$ the rectified linear unit. All of them are entry-wise operations. A matrix is represented by a boldface upper-case letter $\mathbf{X}$, with its ij-th entry denoted as $[\mathbf{X}]_{ij}$. $\|\mathbf{X}\|_F$ is the Frobenius norm of a matrix, $\text{tr}(\mathbf{X})$ and $\det(\mathbf{X})$ the trace and determinant of a square matrix, respectively. We use an underlined boldface capital letter $\underline{\mathbf{X}}$ to represent a three-mode tensor. The ijk-th entry of a tensor is denoted as $[\underline{\mathbf{X}}]_{ijk}$. We further use $\underline{\mathbf{X}}^{[i,:,:]}$, $\underline{\mathbf{X}}^{[:,j,:]}$, and $\underline{\mathbf{X}}^{[:,:,k]}$ to denote the i-th, j-th, and k-th slice along the first, second, and third mode, respectively.

Let $\circ : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ denote the Hadamard product between two vectors, i.e.,

$$[\mathbf{a} \circ \mathbf{b}]_i = [\mathbf{a}]_i \cdot [\mathbf{b}]_i,$$

and $\star : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ the circular correlation,[1] i.e.,

$$[\mathbf{a} \star \mathbf{b}]_i = \sum_{k=0}^{n-1} [\mathbf{a}]_k \cdot [\mathbf{b}]_{(k+i) \bmod n}.$$

For details about these operations, refer to [43], [44].

3 KG EMBEDDING WITH FACTS ALONE

Suppose we are given a KG consisting of n entities and m relations. Facts observed in the KG are stored as a collection of triples $\mathbb{D}^+ = \{(h, r, t)\}$. Each triple is composed of a head entity $h \in \mathbb{E}$, a tail entity $t \in \mathbb{E}$, and a relation $r \in \mathbb{R}$ between them, e.g., (`AlfredHitchcock`, `DirectorOf`, `Psycho`). Here, $\mathbb{E}$ denotes the set of entities, and $\mathbb{R}$ the set of relations. KG embedding aims to embed entities and relations into a low-dimensional continuous vector space, so as to simplify computations on the KG. Most of the currently available techniques use facts stored in the KG to perform the embedding task, enforcing embedding to be compatible with the facts.

A typical KG embedding technique generally consists of three steps: (i) representing entities and relations, (ii) defining a scoring function, and (iii) learning entity and relation representations. The first step specifies the form in which entities and relations are represented in a continuous vector space. Entities are usually represented as vectors, i.e., deterministic points in the vector space [13], [14], [15], [16], [19]. Recent work in [45] further takes into account uncertainties of entities, and models them through multivariate Gaussian distributions. Relations are typically taken as operations in the vector space, which can be represented as vectors [14], [15], matrices [16], [18], tensors [19], multivariate Gaussian distributions [45], or even mixtures of Gaussians [46]. Then, in the second step, a scoring function $f_r(h, t)$ is defined on each fact (h, r, t) to measure its plausibility. Facts observed in the KG tend to have higher scores than those that have not been observed. Finally, to learn those entity and relation representations (i.e., embeddings), the third step solves an optimization problem that maximizes the total plausibility of observed facts (i.e., facts contained in $\mathbb{D}^+$).

We roughly categorize such embedding techniques into two groups: *translational distance models* and *semantic matching models*. The former use distance-based scoring functions, and the latter similarity-based ones. In this section, we first introduce these two groups of embedding techniques, and then discuss the training process for them. After that, we compare these embedding techniques in terms of efficiency and effectiveness.

3.1 Translational Distance Models

Translational distance models exploit distance-based scoring functions. They measure the plausibility of a fact as the distance between the two entities, usually after a translation carried out by the relation.

3.1.1 TransE and Its Extensions

TransE. TransE [14] is the most representative translational distance model. It represents both entities and relations as vectors in the same space, say $\mathbb{R}^d$. Given a fact (h, r, t), the relation is interpreted as a translation vector $\mathbf{r}$ so that the embedded entities $\mathbf{h}$ and $\mathbf{t}$ can be connected by $\mathbf{r}$ with low error, i.e., $\mathbf{h}+\mathbf{r} \approx \mathbf{t}$ when (h, r, t) holds. The intuition here originates from [47], which learns distributed word representations to capture linguistic regularities such as $\texttt{Psycho} - \texttt{AlfredHitchcock} \approx \texttt{Avatar} - \texttt{JamesCameron}$. In multi-relational data, such an analogy holds because of the certain relation of `DirectorOf`, and through this relation we can get $\texttt{AlfredHitchcock} + \texttt{DirectorOf} \approx \texttt{Psycho}$ and $\texttt{JamesCameron} + \texttt{DirectorOf} \approx \texttt{Avatar}$. Fig. 1(a) gives a simple illustration of this idea. The scoring function is then defined as the (negative) distance between $\mathbf{h} + \mathbf{r}$ and $\mathbf{t}$, i.e.,

$$f_r(h, t) = -\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_{1/2}.$$

The score is expected to be large if (h, r, t) holds.

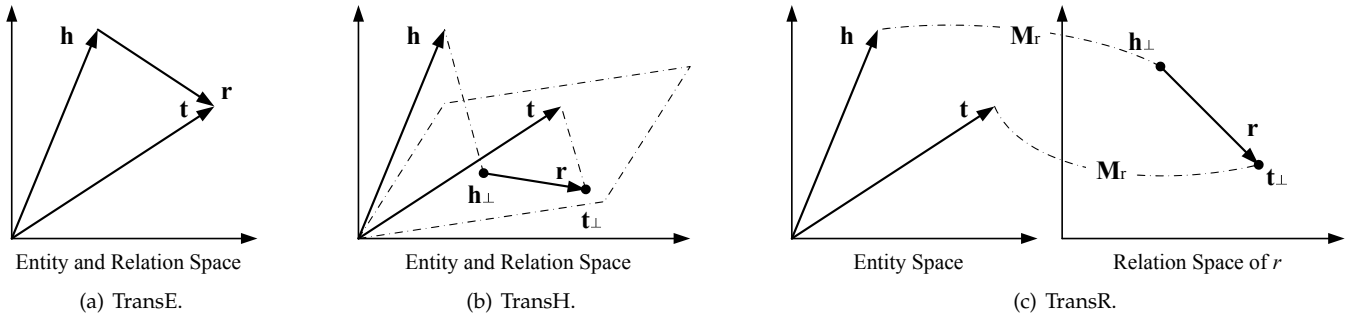
Despite its simplicity and efficiency, TransE has flaws in dealing with 1-to-N, N-to-1, and N-to-N relations [15], [16]. Take 1-to-N relations for example. Given such a relation r, i.e., $\exists i = 1, \cdots, p$ such that $(h, r, t_i) \in \mathbb{D}^+$, TransE enforces $\mathbf{h} + \mathbf{r} \approx \mathbf{t}_i$ for all $i = 1, \cdots, p$, and then $\mathbf{t}_1 \approx \cdots \approx \mathbf{t}_p$. That means, given a 1-to-N relation, e.g., `DirectorOf`, TransE might learn very similar vector representations for `Psycho`, `Rebecca`, and `RearWindow` which are all films directed by `AlfredHitchcock`, even though they are totally different entities. Similar disadvantages exist for N-to-1 and N-to-N relations.

Introducing Relation-Specific Entity Embeddings. To overcome the disadvantages of TransE in dealing with 1-to-N, N-to-1, and N-to-N relations, an effective strategy is to allow an entity to have distinct representations when involved in different relations. In this way, even if the embeddings of `Psycho`, `Rebecca`, and `RearWindow` might be very similar given the relation `DirectorOf`, they could still be far away from each other given other relations.

TransH [15] follows this general idea, by introducing relation-specific hyperplanes. As shown in Fig. 1(b), TransH models entities again as vectors, but each relation r as a

1. For notational brevity, we use zero-indexed vectors.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering



(a) TransE. (b) TransH. (c) TransR.

Fig. 1. Simple illustrations of TransE, TransH, and TransR. The figures are adapted from [15], [16].

vector $\mathbf{r}$ on a hyperplane with $\mathbf{w}_r$ as the normal vector. Given a fact (h, r, t), the entity representations $\mathbf{h}$ and $\mathbf{t}$ are first projected onto the hyperplane, resulting in

$$\mathbf{h}_\perp = \mathbf{h} - \mathbf{w}_r^\top \mathbf{h}\mathbf{w}_r, \quad \mathbf{t}_\perp = \mathbf{t} - \mathbf{w}_r^\top \mathbf{t}\mathbf{w}_r.$$

The projections are then assumed to be connected by $\mathbf{r}$ on the hyperplane with low error if (h, r, t) holds, i.e., $\mathbf{h}_\perp + \mathbf{r} \approx \mathbf{t}_\perp$. The scoring function is accordingly defined as

$$f_r(h, t) = -\|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_2^2,$$

similar to the one used in TransE. By introducing the mechanism of projecting to relation-specific hyperplanes, TransH enables different roles of an entity in different relations.

TransR [16] shares a very similar idea with TransH. But it introduces relation-specific spaces, rather than hyperplanes. In TransR, entities are represented as vectors in an entity space $\mathbb{R}^d$, and each relation is associated with a specific space $\mathbb{R}^k$ and modeled as a translation vector in that space. Given a fact (h, r, t), TransR first projects the entity representations $\mathbf{h}$ and $\mathbf{t}$ into the space specific to relation r, i.e.,

$$\mathbf{h}_\perp = \mathbf{M}_r\mathbf{h}, \quad \mathbf{t}_\perp = \mathbf{M}_r\mathbf{t}.$$

Here $\mathbf{M}_r \in \mathbb{R}^{k\times d}$ is a projection matrix from the entity space to the relation space of r. Then, the scoring function is again defined as

$$f_r(h, t) = -\|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_2^2.$$

Fig. 1(c) gives a simple illustration of TransR. Although powerful in modeling complex relations, TransR introduces a projection matrix for each relation, which requires $\mathcal{O}(dk)$ parameters per relation. So it loses the simplicity and efficiency of TransE/TransH (which model relations as vectors and require only $\mathcal{O}(d)$ parameters per relation). An even more complicated version of the same approach was later proposed in [48], [49]. In this version, each relation is associated with two matrices, one to project head entities and the other tail entities.

TransD [50] simplifies TransR by further decomposing the projection matrix into a product of two vectors. Specifically, for each fact (h, r, t), TransD introduces additional mapping vectors $\mathbf{w}_h, \mathbf{w}_t \in \mathbb{R}^d$ and $\mathbf{w}_r \in \mathbb{R}^k$, along with the entity/relation representations $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$ and $\mathbf{r} \in \mathbb{R}^k$. Two projection matrices $\mathbf{M}_r^1$ and $\mathbf{M}_r^2$ are accordingly defined as

$$\mathbf{M}_r^1 = \mathbf{w}_r\mathbf{w}_h^\top + \mathbf{I}, \quad \mathbf{M}_r^2 = \mathbf{w}_r\mathbf{w}_t^\top + \mathbf{I}.$$

These two projection matrices are then applied on the head entity $\mathbf{h}$ and the tail entity $\mathbf{t}$ respectively to get their projections, i.e.,

$$\mathbf{h}_\perp = \mathbf{M}_r^1\mathbf{h}, \quad \mathbf{t}_\perp = \mathbf{M}_r^2\mathbf{t}.$$

With the projected entities, the scoring function is defined in the same way as in TransR. TransD requires $\mathcal{O}(nd+mk)$ parameters and is more efficient than TransR (which requires $\mathcal{O}(nd + mdk)$ parameters).

TranSparse [51] is another work that simplifies TransR by enforcing sparseness on the projection matrix. It has two versions: TranSparse (share) and TranSparse (separate). The former uses the same sparse projection matrix $\mathbf{M}_r(\theta_r)$ for each relation r, i.e.,

$$\mathbf{h}_\perp = \mathbf{M}_r(\theta_r)\mathbf{h}, \quad \mathbf{t}_\perp = \mathbf{M}_r(\theta_r)\mathbf{t}.$$

The latter introduces two separate sparse projection matrices $\mathbf{M}_r^1(\theta_r^1)$ and $\mathbf{M}_r^2(\theta_r^2)$ for that relation, one to project head entities, and the other tail entities, i.e.,

$$\mathbf{h}_\perp = \mathbf{M}_r^1(\theta_r^1)\mathbf{h}, \quad \mathbf{t}_\perp = \mathbf{M}_r^2(\theta_r^2)\mathbf{t}.$$

Here, θ_r, θ_r^1, and θ_r^2 denote sparseness degrees of these projection matrices. The scoring function is again the same with that used in TransR. By introducing sparse projection matrices, TranSparse reduces the number of parameters to $\mathcal{O}(nd + (1-\theta)mdk)$, where θ is the average sparseness degree of projection matrices.

Relaxing Translational Requirement $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$. Besides allowing entities to have distinct embeddings when involved in different relations, another line of research improves TransE by relaxing the overstrict requirement of $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$. TransM [52] associates each fact (h, r, t) with a weight θ_r specific to the relation, and defines the scoring function as

$$f_r(h, t) = -\theta_r\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_{1/2}.$$

By assigning lower weights to 1-to-N, N-to-1, and N-to-N relations, TransM allows $\mathbf{t}$ to lie farther away from $\mathbf{h} + \mathbf{r}$ in those relations. ManifoldE [53] relaxes $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ to $\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2 \approx \theta_r^2$ for each $(h, r, t) \in \mathbb{D}^+$. As such, $\mathbf{t}$ can lie approximately on a manifold, i.e., a hyper-sphere centered at $\mathbf{h} + \mathbf{r}$ with a radius of θ_r, rather than close to the exact point of $\mathbf{h} + \mathbf{r}$. The scoring function is hence defined as

$$f_r(h, t) = -(\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2 - \theta_r^2)^2.$$

TransF [54] uses a similar idea. Instead of enforcing the strict translation $\mathbf{h}+\mathbf{r} \approx \mathbf{t}$, TransF only requires $\mathbf{t}$ to lie in the same

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

direction with $\mathbf{h}+\mathbf{r}$, and meanwhile $\mathbf{h}$ in the same direction with $\mathbf{t}-\mathbf{r}$. The scoring function then is to match $\mathbf{t}$ with $\mathbf{h}+\mathbf{r}$, and also $\mathbf{h}$ with $\mathbf{t}-\mathbf{r}$, i.e.,

$$f_r(h,t) = (\mathbf{h}+\mathbf{r})^\top \mathbf{t} + (\mathbf{t}-\mathbf{r})^\top \mathbf{h}.$$

TransA [55] introduces for each relation r a symmetric non-negative matrix $\mathbf{M}_r$, and defines the scoring function using an adaptive Mahalanobis distance, i.e.,

$$f_r(h,t) = -(|\mathbf{h}+\mathbf{r}-\mathbf{t}|)^\top \mathbf{M}_r (|\mathbf{h}+\mathbf{r}-\mathbf{t}|).$$

By learning the distance metric $\mathbf{M}_r$, TransA is more flexible in dealing with complex relations.[2]

3.1.2 Gaussian Embeddings

Methods introduced so far model entities as well as relations as deterministic points in vector spaces. Some recent works take into account their uncertainties, and model them as random variables [45], [46]. KG2E [45] represents entities and relations as random vectors drawn from multivariate Gaussian distributions, i.e.,

$$\begin{aligned}\mathbf{h} &\sim \mathcal{N}(\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h),\\ \mathbf{t} &\sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t),\\ \mathbf{r} &\sim \mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r),\end{aligned}$$

where $\boldsymbol{\mu}_h, \boldsymbol{\mu}_t, \boldsymbol{\mu}_r \in \mathbb{R}^d$ are mean vectors, and $\boldsymbol{\Sigma}_h, \boldsymbol{\Sigma}_t, \boldsymbol{\Sigma}_r \in \mathbb{R}^{d\times d}$ covariance matrices. Then, inspired by the translational assumption, KG2E scores a fact by measuring the distance between the two random vectors of $\mathbf{t}-\mathbf{h}$ and $\mathbf{r}$, i.e., the two distributions of $\mathcal{N}(\boldsymbol{\mu}_t-\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_t+\boldsymbol{\Sigma}_h)$ and $\mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)$. Two types of distance measures are used. One is the Kullback-Leibler divergence [58] which defines

$$\begin{aligned}f_r(h,t) &= -\int \mathcal{N}_{\mathbf{x}}(\boldsymbol{\mu}_t-\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_t+\boldsymbol{\Sigma}_h) \ln \frac{\mathcal{N}_{\mathbf{x}}(\boldsymbol{\mu}_t-\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_t+\boldsymbol{\Sigma}_h)}{\mathcal{N}_{\mathbf{x}}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)} d\mathbf{x}\\ &\propto -\mathrm{tr}(\boldsymbol{\Sigma}_r^{-1}(\boldsymbol{\Sigma}_h+\boldsymbol{\Sigma}_t)) - \boldsymbol{\mu}^\top \boldsymbol{\Sigma}_r^{-1}\boldsymbol{\mu} - \ln \frac{\det(\boldsymbol{\Sigma}_r)}{\det(\boldsymbol{\Sigma}_h+\boldsymbol{\Sigma}_t)},\end{aligned}$$

and the other is the probability inner product [59] which introduces

$$\begin{aligned}f_r(h,t) &= \int \mathcal{N}_{\mathbf{x}}(\boldsymbol{\mu}_t-\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_t+\boldsymbol{\Sigma}_h) \cdot \mathcal{N}_{\mathbf{x}}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r) d\mathbf{x}\\ &\propto -\boldsymbol{\mu}^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} - \ln(\det(\boldsymbol{\Sigma})).\end{aligned}$$

Here $\boldsymbol{\mu} = \boldsymbol{\mu}_h + \boldsymbol{\mu}_r - \boldsymbol{\mu}_t$ and $\boldsymbol{\Sigma} = \boldsymbol{\Sigma}_h + \boldsymbol{\Sigma}_r + \boldsymbol{\Sigma}_t$. With the help of Gaussian embeddings, KG2E can effectively model uncertainties of entities and relations in KGs.

TransG [46] also models entities with Gaussian distributions, i.e.,

$$\mathbf{h} \sim \mathcal{N}(\boldsymbol{\mu}_h, \sigma_h^2\mathbf{I}), \quad \mathbf{t} \sim \mathcal{N}(\boldsymbol{\mu}_t, \sigma_t^2\mathbf{I}).$$

But it believes that a relation can have multiple semantics, and hence should be represented as a mixture of Gaussian distributions, i.e.,

$$\mathbf{r} = \sum_i \pi_r^i \boldsymbol{\mu}_r^i, \quad \boldsymbol{\mu}_r^i \sim \mathcal{N}\left(\boldsymbol{\mu}_t - \boldsymbol{\mu}_h, (\sigma_h^2+\sigma_t^2)\mathbf{I}\right).$$

Here, $\boldsymbol{\mu}_r^i$ is the embedding for the i-th semantic, and π_r^i the weight of that semantic. The scoring function is accordingly defined as

$$f_r(h,t) = \sum_i \pi_r^i \exp\left(\frac{-\|\boldsymbol{\mu}_h + \boldsymbol{\mu}_r^i - \boldsymbol{\mu}_t\|_2^2}{\sigma_h^2+\sigma_t^2}\right),$$

which is a mixture of translational distances introduced by different semantics of the relation. These semantic components can be learned automatically from the data using the Chinese restaurant process [60], [61].

3.1.3 Other Distance Models

Unstructured model (UM) [56] is a naive version of TransE by setting all $\mathbf{r}=\mathbf{0}$, leading to a scoring function

$$f_r(h,t) = -\|\mathbf{h}-\mathbf{t}\|_2^2.$$

Obviously, it cannot distinguish different relations. Structured embedding (SE) [57] uses two separate matrices $\mathbf{M}_r^1$ and $\mathbf{M}_r^2$ to project head and tail entities for each relation r, and the score is

$$f_r(h,t) = -\|\mathbf{M}_r^1\mathbf{h} - \mathbf{M}_r^2\mathbf{t}\|_1.$$

Table 1 summarizes entity/relation representations and scoring functions used in these translational distance models. For all the models, there are constraints imposed on them, e.g., enforcing vector embeddings to have, at most, a unit ℓ_2 norm. Some of the constraints are converted into regularization terms during optimization.

3.2 Semantic Matching Models

Semantic matching models exploit similarity-based scoring functions. They measure plausibility of facts by matching latent semantics of entities and relations embodied in their vector space representations.

3.2.1 RESCAL and Its Extensions

RESCAL. RESCAL [13] (a.k.a. the bilinear model [17]) associates each entity with a vector to capture its latent semantics. Each relation is represented as a matrix which models pairwise interactions between latent factors. The score of a fact (h,r,t) is defined by a bilinear function

$$f_r(h,t) = \mathbf{h}^\top \mathbf{M}_r \mathbf{t} = \sum_{i=0}^{d-1}\sum_{j=0}^{d-1} [\mathbf{M}_r]_{ij} \cdot [\mathbf{h}]_i \cdot [\mathbf{t}]_j,$$

where $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$ are vector representations of the entities, and $\mathbf{M}_r \in \mathbb{R}^{d\times d}$ is a matrix associated with the relation.[3] This score captures pairwise interactions between all the components of $\mathbf{h}$ and $\mathbf{t}$ (see also Fig. 2(a)), which requires $\mathcal{O}(d^2)$ parameters per relation. [17] further assumes that all $\mathbf{M}_r$ decompose over a common set of rank-1 matrices, i.e., $\mathbf{M}_r = \sum_i \pi_r^i \mathbf{u}_i \mathbf{v}_i^\top$. TATEC [64] models not only the three-way interaction $\mathbf{h}^\top\mathbf{M}_r\mathbf{t}$ but also two-way interactions, e.g., those between an entity and a relation. The scoring function is $f_r(h,t) = \mathbf{h}^\top\mathbf{M}_r\mathbf{t} + \mathbf{h}^\top\mathbf{r} + \mathbf{t}^\top\mathbf{r} + \mathbf{h}^\top\mathbf{D}\mathbf{t}$, where $\mathbf{D}$ is a diagonal matrix shared across all different relations.[4]

2. To make $f_r(h,t)$ a Mahalanobis distance, $\mathbf{M}_r$ ought to be positive semidefinite. But [55] did not impose this constraint, and only required $\mathbf{M}_r$ to be symmetric and non-negative.

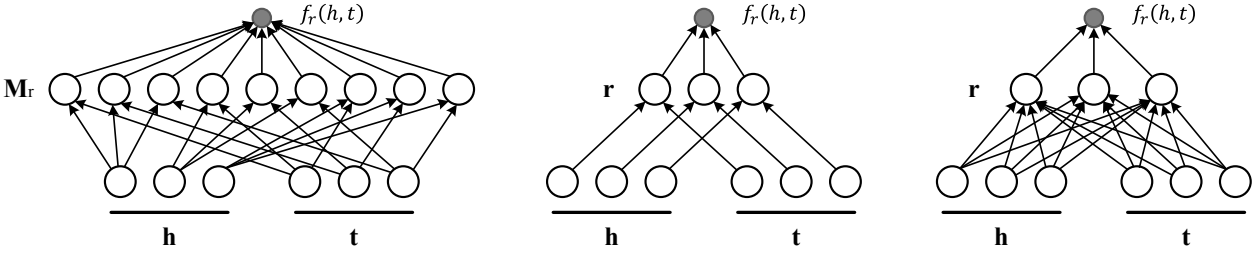
3. Note that we use zero-indexed vectors and matrices for notation brevity in this section.

4. TATEC can actually use different entity representations in the 2-way and 3-way terms. Here we omit this for notation brevity.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

TABLE 1
Summary of Translational Distance Models (See Section 3.1 for Details)

Method	Ent. embedding	Rel. embedding	Scoring function $f_r(h,t)$	Constraints/Regularization
TransE [14]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^d$	$-\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|_{1/2}$	$\|\mathbf{h}\|_2=1, \|\mathbf{t}\|_2=1$
TransH [15]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r},\mathbf{w}_r\in\mathbb{R}^d$	$-\|(\mathbf{h}-\mathbf{w}_r^\top\mathbf{h}\mathbf{w}_r)+\mathbf{r}-(\mathbf{t}-\mathbf{w}_r^\top\mathbf{t}\mathbf{w}_r)\|_2^2$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1$ $\|\mathbf{w}_r^\top\mathbf{r}\|/\|\mathbf{r}\|_2\le\epsilon, \|\mathbf{w}_r\|_2=1$
TransR [16]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^k, \mathbf{M}_r\in\mathbb{R}^{k\times d}$	$-\|\mathbf{M}_r\mathbf{h}+\mathbf{r}-\mathbf{M}_r\mathbf{t}\|_2^2$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{r}\|_2\le 1$ $\|\mathbf{M}_r\mathbf{h}\|_2\le 1, \|\mathbf{M}_r\mathbf{t}\|_2\le 1$
TransD [50]	$\mathbf{h},\mathbf{w}_h\in\mathbb{R}^d$ $\mathbf{t},\mathbf{w}_t\in\mathbb{R}^d$	$\mathbf{r},\mathbf{w}_r\in\mathbb{R}^k$	$-\|(\mathbf{w}_r\mathbf{w}_h^\top+\mathbf{I})\mathbf{h}+\mathbf{r}-(\mathbf{w}_r\mathbf{w}_t^\top+\mathbf{I})\mathbf{t}\|_2^2$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{r}\|_2\le 1$ $\|(\mathbf{w}_r\mathbf{w}_h^\top+\mathbf{I})\mathbf{h}\|_2\le 1$ $\|(\mathbf{w}_r\mathbf{w}_t^\top+\mathbf{I})\mathbf{t}\|_2\le 1$
TranSparse [51]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^k, \mathbf{M}_r(\theta_r)\in\mathbb{R}^{k\times d}$ $\mathbf{M}_r^1(\theta_r^1), \mathbf{M}_r^2(\theta_r^2)\in\mathbb{R}^{k\times d}$	$-\|\mathbf{M}_r(\theta_r)\mathbf{h}+\mathbf{r}-\mathbf{M}_r(\theta_r)\mathbf{t}\|_{1/2}^2$ $-\|\mathbf{M}_r^1(\theta_r^1)\mathbf{h}+\mathbf{r}-\mathbf{M}_r^2(\theta_r^2)\mathbf{t}\|_{1/2}^2$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{r}\|_2\le 1$ $\|\mathbf{M}_r(\theta_r)\mathbf{h}\|_2\le 1, \|\mathbf{M}_r(\theta_r)\mathbf{t}\|_2\le 1$ $\|\mathbf{M}_r^1(\theta_r^1)\mathbf{h}\|_2\le 1, \|\mathbf{M}_r^2(\theta_r^2)\mathbf{t}\|_2\le 1$
TransM [52]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^d$	$-\theta_r\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|_{1/2}$	$\|\mathbf{h}\|_2=1, \|\mathbf{t}\|_2=1$
ManifoldE [53]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^d$	$-(\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|_2^2-\theta_r^2)^2$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{r}\|_2\le 1$
TransF [54]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^d$	$(\mathbf{h}+\mathbf{r})^\top\mathbf{t}+(\mathbf{t}-\mathbf{r})^\top\mathbf{h}$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{r}\|_2\le 1$
TransA [55]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^d, \mathbf{M}_r\in\mathbb{R}^{d\times d}$	$-(\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|)^\top\mathbf{M}_r(\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|)$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{r}\|_2\le 1$ $\|\mathbf{M}_r\|_F\le 1, [\mathbf{M}_r]_{ij}=[\mathbf{M}_r]_{ji}\ge 0$
KG2E [45]	$\mathbf{h}\sim\mathcal{N}(\boldsymbol{\mu}_h,\boldsymbol{\Sigma}_h)$ $\mathbf{t}\sim\mathcal{N}(\boldsymbol{\mu}_t,\boldsymbol{\Sigma}_t)$ $\boldsymbol{\mu}_h,\boldsymbol{\mu}_t\in\mathbb{R}^d$ $\boldsymbol{\Sigma}_h,\boldsymbol{\Sigma}_t\in\mathbb{R}^{d\times d}$	$\mathbf{r}\sim\mathcal{N}(\boldsymbol{\mu}_r,\boldsymbol{\Sigma}_r)$ $\boldsymbol{\mu}_r\in\mathbb{R}^d, \boldsymbol{\Sigma}_r\in\mathbb{R}^{d\times d}$	$-\mathrm{tr}(\boldsymbol{\Sigma}_r^{-1}(\boldsymbol{\Sigma}_h+\boldsymbol{\Sigma}_t))-\boldsymbol{\mu}^\top\boldsymbol{\Sigma}_r^{-1}\boldsymbol{\mu}-\ln\frac{\det(\boldsymbol{\Sigma}_r)}{\det(\boldsymbol{\Sigma}_h+\boldsymbol{\Sigma}_t)}$ $-\boldsymbol{\mu}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}-\ln(\det(\boldsymbol{\Sigma}))$ $\boldsymbol{\mu}=\boldsymbol{\mu}_h+\boldsymbol{\mu}_r-\boldsymbol{\mu}_t$ $\boldsymbol{\Sigma}=\boldsymbol{\Sigma}_h+\boldsymbol{\Sigma}_r+\boldsymbol{\Sigma}_t$	$\|\boldsymbol{\mu}_h\|_2\le 1, \|\boldsymbol{\mu}_t\|_2\le 1, \|\boldsymbol{\mu}_r\|_2\le 1$ $c_{min}\mathbf{I}\le\boldsymbol{\Sigma}_h\le c_{max}\mathbf{I}$ $c_{min}\mathbf{I}\le\boldsymbol{\Sigma}_t\le c_{max}\mathbf{I}$ $c_{min}\mathbf{I}\le\boldsymbol{\Sigma}_r\le c_{max}\mathbf{I}$
TransG [46]	$\mathbf{h}\sim\mathcal{N}(\boldsymbol{\mu}_h,\sigma_h^2\mathbf{I})$ $\mathbf{t}\sim\mathcal{N}(\boldsymbol{\mu}_t,\sigma_t^2\mathbf{I})$ $\boldsymbol{\mu}_h,\boldsymbol{\mu}_t\in\mathbb{R}^d$	$\boldsymbol{\mu}_r^i\sim\mathcal{N}(\boldsymbol{\mu}_t-\boldsymbol{\mu}_h,(\sigma_h^2+\sigma_t^2)\mathbf{I})$ $\mathbf{r}=\sum_i\pi_r^i\boldsymbol{\mu}_r^i\in\mathbb{R}^d$	$\sum_i\pi_r^i\exp\left(-\frac{\|\boldsymbol{\mu}_h+\boldsymbol{\mu}_r^i-\boldsymbol{\mu}_t\|_2^2}{\sigma_h^2+\sigma_t^2}\right)$	$\|\boldsymbol{\mu}_h\|_2\le 1, \|\boldsymbol{\mu}_t\|_2\le 1, \|\boldsymbol{\mu}_r^i\|_2\le 1$
UM [56]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	——	$-\|\mathbf{h}-\mathbf{t}\|_2^2$	$\|\mathbf{h}\|_2=1, \|\mathbf{t}\|_2=1$
SE [57]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{M}_r^1,\mathbf{M}_r^2\in\mathbb{R}^{d\times d}$	$-\|\mathbf{M}_r^1\mathbf{h}-\mathbf{M}_r^2\mathbf{t}\|_1$	$\|\mathbf{h}\|_2=1, \|\mathbf{t}\|_2=1$



(a) RESCAL. (b) DistMult. (c) HolE.

Fig. 2. Simple illustrations of RESCAL, DistMulti, and HolE. The figures are adapted from [62].

DistMult. DistMult [65] simplifies RESCAL by restricting $\mathbf{M}_r$ to diagonal matrices. For each relation r, it introduces a vector embedding $\mathbf{r}\in\mathbb{R}^d$ and requires $\mathbf{M}_r=\mathrm{diag}(\mathbf{r})$. The scoring function is hence defined as

$$f_r(h,t)=\mathbf{h}^\top\mathrm{diag}(\mathbf{r})\mathbf{t}=\sum_{i=0}^{d-1}[\mathbf{r}]_i\cdot[\mathbf{h}]_i\cdot[\mathbf{t}]_i.$$

This score captures pairwise interactions between only the components of $\mathbf{h}$ and $\mathbf{t}$ along the same dimension (see also Fig. 2(b)), and reduces the number of parameters to $\mathcal{O}(d)$ per relation. However, since $\mathbf{h}^\top\mathrm{diag}(\mathbf{r})\mathbf{t}=\mathbf{t}^\top\mathrm{diag}(\mathbf{r})\mathbf{h}$ for any $\mathbf{h}$ and $\mathbf{t}$, this over-simplified model can only deal with symmetric relations which is clearly not powerful enough for general KGs.

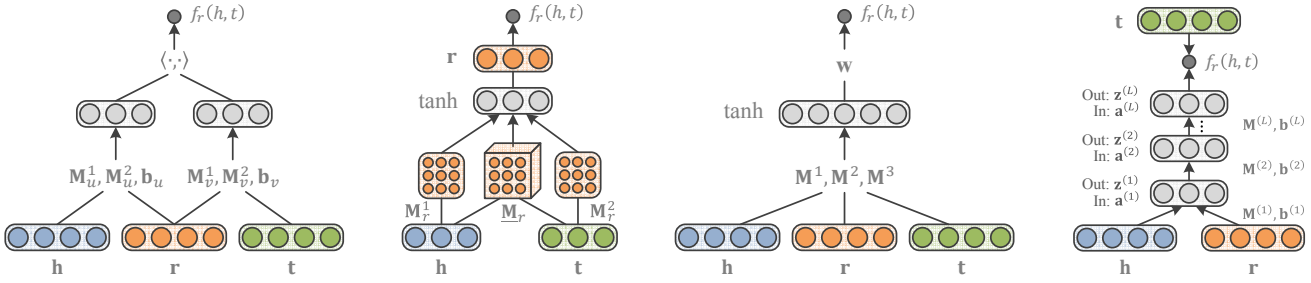
Holographic Embeddings (HolE). HolE [62] combines the expressive power of RESCAL with the efficiency and simplicity of DistMult. It represents both entities and relations as vectors in $\mathbb{R}^d$. Given a fact (h,r,t), the entity representations are first composed into $\mathbf{h}\star\mathbf{t}\in\mathbb{R}^d$ by using the circular correlation operation [44], namely

$$[\mathbf{h}\star\mathbf{t}]_i=\sum_{k=0}^{d-1}[\mathbf{h}]_k\cdot[\mathbf{t}]_{(k+i)\bmod d}.$$

The compositional vector is then matched with the relation representation to score that fact, i.e.,

$$f_r(h,t)=\mathbf{r}^\top(\mathbf{h}\star\mathbf{t})=\sum_{i=0}^{d-1}[\mathbf{r}]_i\sum_{k=0}^{d-1}[\mathbf{h}]_k\cdot[\mathbf{t}]_{(k+i)\bmod d}.$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering



(a) SME. (b) NTN. (c) MLP. (d) NAM.

Fig. 3. Neural network architectures of SME, NTN, MLP, and NAM. The figures are adapted from [18], [19], [63].

Circular correlation makes a compression of pairwise interactions (see also Fig. 2(c)). So HolE requires only $\mathcal{O}(d)$ parameters per relation, which is more efficient than RESCAL. Meanwhile, since circular correlation is not commutative, i.e., $\mathbf{h}\star\mathbf{t} \neq \mathbf{t}\star\mathbf{h}$, HolE is able to model asymmetric relations as RESCAL does.

Complex Embeddings (ComplEx). ComplEx [66] extends DistMult by introducing complex-valued embeddings so as to better model asymmetric relations. In ComplEx, entity and relation embeddings $\mathbf{h}, \mathbf{r}, \mathbf{t}$ no longer lie in a real space but a complex space, say $\mathbb{C}^d$. The score of a fact (h, r, t) is defined as

$$f_r(h,t) = \mathrm{Re}\big(\mathbf{h}^\top \mathrm{diag}(\mathbf{r})\bar{\mathbf{t}}\big) = \mathrm{Re}\big(\sum_{i=0}^{d-1}[\mathbf{r}]_i \cdot [\mathbf{h}]_i \cdot [\bar{\mathbf{t}}]_i\big),$$

where $\bar{\mathbf{t}}$ is the conjugate of $\mathbf{t}$ and $\mathrm{Re}(\cdot)$ means taking the real part of a complex value. This scoring function is not symmetric any more, and facts from asymmetric relations can receive different scores depending on the order of entities involved. [67] recently showed that every ComplEx has an equivalent HolE, and conversely, HolE is subsumed by ComplEx as a special case in which the conjugate symmetry is imposed on embeddings.

ANALOGY. ANALOGY [68] extends RESCAL so as to further model the analogical properties of entities and relations, e.g., "`AlfredHitchcock` is to `Psycho` as `JamesCameron` is to `Avatar`". It follows RESCAL and employs a bilinear scoring function

$$f_r(h,t) = \mathbf{h}^\top \mathbf{M}_r \mathbf{t},$$

where $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$ are vector embeddings for the entities, and $\mathbf{M}_r \in \mathbb{R}^{d\times d}$ is a linear map associated with the relation. To model the analogical structures, it further requires relations' linear maps to be normal and mutually commutative, i.e.,

$$\begin{aligned}\text{normality:}\quad & \mathbf{M}_r\mathbf{M}_r^\top = \mathbf{M}_r^\top\mathbf{M}_r,\ \forall r \in \mathbb{R};\\ \text{commutativity:}\quad & \mathbf{M}_r\mathbf{M}_{r'} = \mathbf{M}_{r'}\mathbf{M}_r,\ \forall r, r' \in \mathbb{R}.\end{aligned}$$

Although ANALOGY represents relations as matrices, these matrices can be simultaneously block-diagonalized into a set of sparse almost-diagonal matrices, each of which consists of only $\mathcal{O}(d)$ free parameters. It has been shown that the previously introduced methods of DistMult, HolE, and ComplEx can all be subsumed by ANALOGY as special cases in a principled manner.

3.2.2 Matching with Neural Networks

Semantic Matching Energy (SME). SME [18] conducts semantic matching using neural network architectures. Given a fact (h, r, t), it first projects entities and relations to their vector embeddings in the input layer. The relation $\mathbf{r}$ is then combined with the head entity $\mathbf{h}$ to get $g_u(\mathbf{h}, \mathbf{r})$, and with the tail entity $\mathbf{t}$ to get $g_v(\mathbf{t}, \mathbf{r})$ in the hidden layer. The score of a fact is finally defined as matching g_u and g_v by their dot product, i.e.,

$$f_r(h,t) = g_u(\mathbf{h},\mathbf{r})^\top g_v(\mathbf{t},\mathbf{r}).$$

There are two versions of SME: a linear version as well as a bilinear version. SME (linear) defines

$$\begin{aligned}g_u(\mathbf{h},\mathbf{r}) &= \mathbf{M}_u^1\mathbf{h} + \mathbf{M}_u^2\mathbf{r} + \mathbf{b}_u,\\ g_v(\mathbf{t},\mathbf{r}) &= \mathbf{M}_v^1\mathbf{t} + \mathbf{M}_v^2\mathbf{r} + \mathbf{b}_v,\end{aligned}$$

while SME (bilinear) defines

$$\begin{aligned}g_u(\mathbf{h},\mathbf{r}) &= (\mathbf{M}_u^1\mathbf{h}) \circ (\mathbf{M}_u^2\mathbf{r}) + \mathbf{b}_u,\\ g_v(\mathbf{t},\mathbf{r}) &= (\mathbf{M}_v^1\mathbf{t}) \circ (\mathbf{M}_v^2\mathbf{r}) + \mathbf{b}_v.\end{aligned}$$

Here, $\mathbf{M}_u^1, \mathbf{M}_u^2, \mathbf{M}_v^1, \mathbf{M}_v^2 \in \mathbb{R}^{d\times d}$ are weight matrices and $\mathbf{b}_u, \mathbf{b}_v \in \mathbb{R}^d$ bias vectors shared across different relations.[5] Fig. 3(a) provides a simple illustration of SME.

Neural Tensor Network (NTN). NTN [19] is another neural network architecture. Given a fact, it first projects entities to their vector embeddings in the input layer. Then, the two entities $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$ are combined by a relation-specific tensor $\underline{\mathbf{M}}_r \in \mathbb{R}^{d\times d\times k}$ (along with other parameters) and mapped to a non-linear hidden layer. Finally, a relation-specific linear output layer gives the score

$$f_r(h,t) = \mathbf{r}^\top \tanh(\mathbf{h}^\top\underline{\mathbf{M}}_r\mathbf{t} + \mathbf{M}_r^1\mathbf{h} + \mathbf{M}_r^2\mathbf{t} + \mathbf{b}_r),$$

where $\mathbf{M}_r^1, \mathbf{M}_r^2 \in \mathbb{R}^{k\times d}$ and $\mathbf{b}_r \in \mathbb{R}^k$ are relation-specific weight matrices and bias vectors, respectively. The bilinear tensor product $\mathbf{h}^\top\underline{\mathbf{M}}_r\mathbf{t}$ results in a vector, with the i-th entry computed as $\mathbf{h}^\top\underline{\mathbf{M}}_r^{[:,:,i]}\mathbf{t}$. Fig. 3(b) gives a simple illustration of NTN. By setting all $\underline{\mathbf{M}}_r = \mathbf{0}$ and $\mathbf{b}_r = \mathbf{0}$, NTN degenerates to the single layer model (SLM) [19]. NTN might be the most expressive model to date. But it requires $\mathcal{O}(d^2k)$ parameters per relation, and is not sufficiently simple and efficient to handle large-scale KGs.

5. In [18], SME (bilinear) is defined with three-mode tensors instead of matrices. The two forms are equivalent [70]. Note also that $\mathbf{M}_u^1$, $\mathbf{M}_u^2$, $\mathbf{M}_v^1$, $\mathbf{M}_v^2$ are not necessarily square matrices.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

TABLE 2
Summary of Semantic Matching Models (See Section 3.2 for Details)

Method	Ent. embedding	Rel. embedding	Scoring function $f_r(h,t)$	Constraints/Regularization
RESCAL [13]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{M}_r\in\mathbb{R}^{d\times d}$	$\mathbf{h}^\top\mathbf{M}_r\mathbf{t}$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{M}_r\|_F\le 1$ $\mathbf{M}_r=\sum_i \pi_r^i\mathbf{u}_i\mathbf{v}_i^\top$ (required in [17])
TATEC [64]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^d, \mathbf{M}_r\in\mathbb{R}^{d\times d}$	$\mathbf{h}^\top\mathbf{M}_r\mathbf{t}+\mathbf{h}^\top\mathbf{r}+\mathbf{t}^\top\mathbf{r}+\mathbf{h}^\top\mathbf{D}\mathbf{t}$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{r}\|_2\le 1$ $\|\mathbf{M}_r\|_F\le 1$
DistMult [65]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^d$	$\mathbf{h}^\top\mathrm{diag}(\mathbf{r})\mathbf{t}$	$\|\mathbf{h}\|_2=1, \|\mathbf{t}\|_2=1, \|\mathbf{r}\|_2\le 1$
HolE [62]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^d$	$\mathbf{r}^\top(\mathbf{h}\star\mathbf{t})$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{r}\|_2\le 1$
ComplEx [66]	$\mathbf{h},\mathbf{t}\in\mathbb{C}^d$	$\mathbf{r}\in\mathbb{C}^d$	$\mathrm{Re}\big(\mathbf{h}^\top\mathrm{diag}(\mathbf{r})\bar{\mathbf{t}}\big)$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{r}\|_2\le 1$
ANALOGY [68]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{M}_r\in\mathbb{R}^{d\times d}$	$\mathbf{h}^\top\mathbf{M}_r\mathbf{t}$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{M}_r\|_F\le 1$ $\mathbf{M}_r\mathbf{M}_r^\top=\mathbf{M}_r^\top\mathbf{M}_r$ $\mathbf{M}_r\mathbf{M}_{r'}=\mathbf{M}_{r'}\mathbf{M}_r$
SME [18]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^d$	$(\mathbf{M}_u^1\mathbf{h}+\mathbf{M}_u^2\mathbf{r}+\mathbf{b}_u)^\top(\mathbf{M}_v^1\mathbf{t}+\mathbf{M}_v^2\mathbf{r}+\mathbf{b}_v)$ $\big((\mathbf{M}_u^1\mathbf{h})\circ(\mathbf{M}_u^2\mathbf{r})+\mathbf{b}_u\big)^\top\big((\mathbf{M}_v^1\mathbf{t})\circ(\mathbf{M}_v^2\mathbf{r})+\mathbf{b}_v\big)$	$\|\mathbf{h}\|_2=1, \|\mathbf{t}\|_2=1$
NTN [19]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r},\mathbf{b}_r\in\mathbb{R}^k, \underline{\mathbf{M}}_r\in\mathbb{R}^{d\times d\times k}$ $\mathbf{M}_r^1,\mathbf{M}_r^2\in\mathbb{R}^{k\times d}$	$\mathbf{r}^\top\tanh(\mathbf{h}^\top\underline{\mathbf{M}}_r\mathbf{t}+\mathbf{M}_r^1\mathbf{h}+\mathbf{M}_r^2\mathbf{t}+\mathbf{b}_r)$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{r}\|_2\le 1$ $\|\mathbf{b}_r\|_2\le 1, \|\underline{\mathbf{M}}_r^{[:,:,i]}\|_F\le 1$ $\|\mathbf{M}_r^1\|_F\le 1, \|\mathbf{M}_r^2\|_F\le 1$
SLM [19]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^k, \mathbf{M}_r^1,\mathbf{M}_r^2\in\mathbb{R}^{k\times d}$	$\mathbf{r}^\top\tanh(\mathbf{M}_r^1\mathbf{h}+\mathbf{M}_r^2\mathbf{t})$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{r}\|_2\le 1$ $\|\mathbf{M}_r^1\|_F\le 1, \|\mathbf{M}_r^2\|_F\le 1$
MLP [69]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^d$	$\mathbf{w}^\top\tanh(\mathbf{M}^1\mathbf{h}+\mathbf{M}^2\mathbf{r}+\mathbf{M}^3\mathbf{t})$	$\|\mathbf{h}\|_2\le 1, \|\mathbf{t}\|_2\le 1, \|\mathbf{r}\|_2\le 1$
NAM [63]	$\mathbf{h},\mathbf{t}\in\mathbb{R}^d$	$\mathbf{r}\in\mathbb{R}^d$	$f_r(h,t)=\mathbf{t}^\top\mathbf{z}^{(L)}$ $\mathbf{z}^{(\ell)}=\mathrm{ReLU}(\mathbf{a}^{(\ell)}),\ \ \mathbf{a}^{(\ell)}=\mathbf{M}^{(\ell)}\mathbf{z}^{(\ell-1)}+\mathbf{b}^{(\ell)}$ $\mathbf{z}^{(0)}=[\mathbf{h};\mathbf{r}]$	——

Multi-Layer Perceptron (MLP). MLP [69] is a simpler approach where each relation (as well as entity) is associated with a single vector. As illustrated in Fig. 3(c), given a fact (h,r,t), the vector embeddings $\mathbf{h}$, $\mathbf{r}$, and $\mathbf{t}$ are concatenated in the input layer, and mapped to a non-linear hidden layer. The score is then generated by a linear output layer, i.e.,

$$f_r(h,t)=\mathbf{w}^\top\tanh(\mathbf{M}^1\mathbf{h}+\mathbf{M}^2\mathbf{r}+\mathbf{M}^3\mathbf{t}).$$

Here, $\mathbf{M}^1,\mathbf{M}^2,\mathbf{M}^3\in\mathbb{R}^{d\times d}$ are the first layer weights, and $\mathbf{w}\in\mathbb{R}^d$ the second layer weights, all shared across different relations.[6]

Neural Association Model (NAM). NAM [63] conducts semantic matching with a "deep" architecture. Given a fact (h,r,t), it first concatenates the vector embeddings of the head entity and the relation in the input layer, which gives $\mathbf{z}^{(0)}=[\mathbf{h};\mathbf{r}]\in\mathbb{R}^{2d}$. The input $\mathbf{z}^{(0)}$ is then fed into a deep neural network consisting of L rectified linear hidden layers such that

$$\mathbf{a}^{(\ell)}=\mathbf{M}^{(\ell)}\mathbf{z}^{(\ell-1)}+\mathbf{b}^{(\ell)},\quad \ell=1,\cdots,L,$$
$$\mathbf{z}^{(\ell)}=\mathrm{ReLU}(\mathbf{a}^{(\ell)}),\quad \ell=1,\cdots,L,$$

where $\mathbf{M}^{(\ell)}$ and $\mathbf{b}^{(\ell)}$ represent the weight matrix and bias for the ℓ-th layer respectively. After the feedforward process, the score is given by matching the output of the last hidden layer and the embedding of the tail entity, i.e.,

$$f_r(h,t)=\mathbf{t}^\top\mathbf{z}^{(L)}.$$

Fig. 3(d) provides a simple illustration of NAM. It has a more complicated version which connects the relation embedding $\mathbf{r}$ to all hidden layers in the network. Table 2 summarizes entity/relation representations and scoring functions used in these semantic matching models. Constraints imposed on these models are also presented.[7]

3.3 Model Training

This section discusses routine training procedures for KG embedding models. We consider two widely used assumptions: the *open world assumption* and *closed world assumption*.

3.3.1 Training under Open World Assumption

The open world assumption (OWA) states that KGs contain only true facts and non-observed facts can be either false or just missing [72]. In this case, $\mathbb{D}^+$ stores only positive examples. Negative examples can be generated by heuristics such as the local closed world assumption [69] (detailed later in this section). Given the positive set $\mathbb{D}^+$ and a negative set $\mathbb{D}^-$ constructed accordingly, we can learn entity and relation representations Θ by minimizing, for instance, the logistic loss, i.e.,

$$\min_{\Theta}\sum_{\tau\in\mathbb{D}^+\cup\mathbb{D}^-}\log\left(1+\exp(-y_{hrt}\cdot f_r(h,t))\right),\qquad(1)$$

where $\tau=(h,r,t)$ is a training example in $\mathbb{D}^+\cup\mathbb{D}^-$, and $y_{hrt}=\pm 1$ the label (positive or negative) of that example. It has been shown that minimizing the logistic loss can help

6. $\mathbf{M}^1$, $\mathbf{M}^2$, $\mathbf{M}^3$ are not necessarily square matrices.

7. Instead of imposing constraints or regularization terms, NAM adopts the dropout approach [71] during training to avoid over-fitting.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

to find compact representations for some complex relational patterns such as transitive relations [73]. Besides the logistic loss, we can also use a pairwise ranking loss such as

$$\min_{\Theta} \sum_{\tau^+ \in \mathbb{D}^+} \sum_{\tau^- \in \mathbb{D}^-} \max\left(0, \gamma - f_r(h,t) + f_{r'}(h',t')\right) \quad (2)$$

to make the scores of positive facts higher than those of negative ones.[8] Here, $\tau^+ = (h,r,t)$ is a positive example, $\tau^- = (h',r',t')$ a negative one, and γ a margin separating them. Minimizing the pairwise ranking loss has an additional advantage: it does not assume that negative examples are necessarily false, just that they are more invalid than those positive ones [36]. Note that in both types of optimization, there are constraints and/or regularization terms specified by different embedding models (see Table 1 and Table 2 for details). Trouillon et al. [66] have shown that the logistic loss generally yields better results for semantic matching models such as DistMult and ComplEx, while the pairwise ranking loss may be more suitable for translational distance models like TransE.

The optimization Eq. (1) and Eq. (2) can be easily carried out by stochastic gradient descent (SGD) [74] in minibatch mode. After initializing entity and relation embeddings, at each iteration, a small set of positive facts is sampled from $\mathbb{D}^+$, and for each positive fact, one or more negative facts are generated accordingly. These positive and negative facts then serve as training examples in a minibatch. After the minibatch, embeddings are updated by a gradient step with constant or adaptive learning rates. AdaGrad [75] is usually used to tune the learning rate. Algorithm 1 summarizes this training procedure, where a single negative is generated for each positive fact. Next, we discuss the initialization step (line 1) and negative example generation step (line 6).

Algorithm 1 Training under Open World Assumption

Input: Observed facts $\mathbb{D}^+ = \{(h,r,t)\}$
1: Initialize entity and relation embeddings
2: **loop**
3: $\quad \mathbb{P} \leftarrow$ a small set of positive facts sampled from $\mathbb{D}^+$
4: $\quad \mathbb{B}^+ \leftarrow \emptyset, \ \mathbb{B}^- \leftarrow \emptyset$
5: $\quad$ **foreach** $\tau^+ = (h,r,t) \in \mathbb{P}$ **do**
6: $\quad\quad$ Generate a negative fact $\tau^- = (h',r',t')$
7: $\quad\quad \mathbb{B}^+ \leftarrow \mathbb{B}^+ \cup \{\tau^+\}, \ \mathbb{B}^- \leftarrow \mathbb{B}^- \cup \{\tau^-\}$
8: $\quad$ **end for**
9: $\quad$ Update entity and relation embeddings w.r.t. the gradients of $\sum_{\tau \in \mathbb{B}^+ \cup \mathbb{B}^-} \log\left(1 + \exp(-y_{hrt} \cdot f_r(h,t))\right)$ or $\sum_{\tau^+ \in \mathbb{B}^+, \tau^- \in \mathbb{B}^-} \max\left(0, \gamma - f_r(h,t) + f_{r'}(h',t')\right)$
10: $\quad$ Handle additional constraints or regularization terms
11: **end loop**
Output: Entity and relation embeddings

Initializing Entity and Relation Embeddings. Embeddings for entities and relations are usually initialized randomly from uniform distributions [14] or Gaussian distributions [66]. Some complicated models also initialize their embeddings with results of simple models such as TransE [16]. Another solution is to represent an entity as the average word vector of its name [19] or description [76], and initialize word vectors with those pre-trained on a text corpus.

Generating Negative Training Examples. Given a positive fact $\tau^+ = (h,r,t) \in \mathbb{D}^+$, negative facts can be generated by replacing either the head h or the tail t with a random entity sampled uniformly from $\mathbb{E}$ [14], i.e.,

$$\begin{aligned}\mathbb{D}^- = &\{(h',r,t) | h' \in \mathbb{E} \wedge h' \neq h \wedge (h,r,t) \in \mathbb{D}^+\} \\ &\cup \{(h,r,t') | t' \in \mathbb{E} \wedge t' \neq t \wedge (h,r,t) \in \mathbb{D}^+\}.\end{aligned}$$

Sometimes, negative facts can also be generated by randomly corrupting the relation [18], [45], i.e.,

$$\begin{aligned}\mathbb{D}^- = &\{(h',r,t) | h' \in \mathbb{E} \wedge h' \neq h \wedge (h,r,t) \in \mathbb{D}^+\} \\ &\cup \{(h,r,t') | t' \in \mathbb{E} \wedge t' \neq t \wedge (h,r,t) \in \mathbb{D}^+\} \\ &\cup \{(h,r',t) | r' \in \mathbb{R} \wedge r' \neq r \wedge (h,r,t) \in \mathbb{D}^+\}.\end{aligned}$$

However, this way of uniformly sampling might introduce false-negative training examples, e.g., (`AlfredHitchcock`, `Gender`, `Male`) might be a false-negative example generated for (`JamesCameron`, `Gender`, `Male`) by replacing the head.

To reduce such false-negative examples, [15] proposed to set different probabilities for replacing the head and the tail, i.e., to give more chance to replacing the head if the relation is 1-to-N and the tail if the relation is N-to-1. Consider, for example, the relation `Gender`. For triples from this relation, replacing the tail is obviously more likely to generate true-negative examples. Specifically, given a relation and all its positive facts, [15] first calculates (i) the average number of tail entities per head (denoted as tph), and (ii) the average number of head entities per tail (denoted as hpt). Then, for any positive fact from that relation, [15] corrupts the fact by replacing the head with probability $tph/(tph+hpt)$, and the tail with probability $hpt/(tph+hpt)$. [77] adopted a different strategy to generate negative facts. Given a positive fact, it corrupts a position (i.e. head or tail) using only entities that have appeared in that position with the same relation. That means, given (`JamesCameron`, `Gender`, `Male`), it could generate a negative fact (`JamesCameron`, `Gender`, `Female`) but never (`JamesCameron`, `Gender`, `Avatar`), as `Avatar` never appears as a tail entity of the relation.

Trouillon et al. [66] further investigated the influence of the number of negative facts generated for each positive one. Their study showed that generating more negatives usually leads to better results, and 50 negatives per positive example is a good trade-off between accuracy and training time.

3.3.2 Training under Closed World Assumption

The closed world assumption (CWA) assumes that all facts that are not contained in $\mathbb{D}^+$ are false. In this case, we can learn entity and relation representations Θ by minimizing, for instance, the squared loss, i.e.,

$$\min_{\Theta} \sum_{h,t \in \mathbb{E}, r \in \mathbb{R}} \left(y_{hrt} - f_r(h,t)\right)^2, \quad (3)$$

where $y_{hrt} = 1$ if $(h,r,t) \in \mathbb{D}^+$, and $y_{hrt} = 0$ otherwise. The squared loss sums over all $h,t \in \mathbb{E}$ and $r \in \mathbb{R}$, enforcing observed facts to have scores close to 1, while non-observed facts scores close to 0. Other possible loss functions include the logistic loss [78] and absolute loss [79], [80].

Minimizing the squared loss amounts to factorization of a three-mode tensor represented by the KG. For example,

8. Sometimes a sigmoid function is imposed on $f_r(h,t)$ to make the score a probability within the range of $(0,1)$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

using the squared loss, the optimization for RESCAL becomes a tensor factorization (or collective matrix factorization) problem

$$\min_{\mathbf{E},\{\mathbf{M}_r\}} \sum_{r\in\mathbb{R}} \|\underline{\mathbf{Y}}^{[:,:,r]} - \mathbf{E}\mathbf{M}_r\mathbf{E}^\top\|_F^2 + \lambda_1\|\mathbf{E}\|_F^2 + \lambda_2 \sum_{r\in\mathbb{R}} \|\mathbf{M}_r\|_F^2.$$

Here, $\underline{\mathbf{Y}} \in \{0,1\}^{n\times n\times m}$ is a three-mode tensor encoding the KG whose entries are set such that $[\underline{\mathbf{Y}}]_{hrt} = 1$ if and only if $(h,r,t) \in \mathbb{D}^+$; $\underline{\mathbf{Y}}^{[:,:,r]} \in \{0,1\}^{n\times n}$ is the slice holding all facts from relation r; $\mathbf{E} \in \mathbb{R}^{n\times d}$ stores entity embeddings where each row is a vector representation of an entity; $\mathbf{M}_r \in \mathbb{R}^{d\times d}$ holds the embedding for relation r; and $\lambda_1, \lambda_2 \geq 0$ are regularization coefficients. This optimization problem can be solved efficiently via an alternating least squares (ALS) algorithm, which alternately fixes $\{\mathbf{M}_r\}$ to update $\mathbf{E}$ and then fixes $\mathbf{E}$ to update $\{\mathbf{M}_r\}$. Besides RESCAL, other tensor factorization models such as the CANDECOMP/PARAFAC decomposition [81] and Tucker decomposition [82] have also been applied on KGs to model multi-relational data [83], [84], [85]. For more details about tensor representation of KGs and tensor factorization models, refer to [13], [36] and references therein.

The closed world assumption has several disadvantages. First, it will not hold for incomplete KGs where a lot of true facts are missing from observed data. However, despite their seemingly huge size, most KGs (e.g., Freebase, DBpedia, and YAGO) are highly incomplete [86], making the open world assumption a better choice in this case. In fact, it has been shown that CWA-based models generally perform worse than OWA-based ones in downstream tasks [87]. And [62] has also demonstrated that the CWA-based RESCAL model can perform substantially better if trained under the open world assumption, i.e., minimizing the pairwise ranking loss defined in Eq. (2) rather than the squared loss defined in Eq. (3). Second, the closed world assumption will introduce a huge number of negative examples, which might lead to scalability issues in model training.

3.4 Model Comparison

Table 3 compares space and time complexity of the different models we have discussed. Here, n and m are the number of entities and relations respectively; d and k the dimensionality of entity and relation embedding space respectively (we usually have $d = k$); θ in TranSparse the average sparseness degree of projection matrices; c in TransG the average number of semantic components per relation; and L in NAM the total number of hidden layers in the network. See Section 3.1 and Section 3.2 for details. During the analysis we assume that $d, k \ll n$ and all the models are trained under the open world assumption. We can draw the following conclusions. First, models which represent entities and relations as vectors (e.g., TransE, TransH, DistMult, and ComplEx) are more efficient. They usually have space and time complexity that scales linearly with d. HolE is an exception, as it computes circular correlation via the discrete Fourier transform whose time complexity is $\mathcal{O}(d \log d)$. Second, models which represent relations as matrices (e.g., TransR, SE, and RESCAL) or tensors (e.g., NTN) usually have higher complexity in both space and time, scaling quadratically or cubically with the dimensionality of embedding space. Here ANALOGY is an exception, since the relational matrices can be block-diagonalized into a set of sparse almost-diagonal matrices, each of which consists of only $\mathcal{O}(d)$ free parameters. Finally, models based on neural network architectures (e.g., SME, NTN, MLP, and NAM) generally have higher complexity in time, if not in space, since matrix or even tensor computations are often required in these models.

TABLE 3
Comparison of Models in Space and Time Complexity

Method	Space complexity	Time complexity
TransE [14]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
TransH [15]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
TransR [16]	$\mathcal{O}(nd + mdk)$	$\mathcal{O}(dk)$
TransD [50]	$\mathcal{O}(nd + mk)$	$\mathcal{O}(\max(d,k))$
TranSparse [51]	$\mathcal{O}(nd + (1-\theta)mdk)$	$\mathcal{O}(dk)$
TransM [52]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
ManifoldE [53]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
TransF [54]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
TransA [55]	$\mathcal{O}(nd + md^2)$	$\mathcal{O}(d^2)$
KG2E [45]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
TransG [46]	$\mathcal{O}(nd + mdc)$	$\mathcal{O}(dc)$
UM [56]	$\mathcal{O}(nd)$	$\mathcal{O}(d)$
SE [57]	$\mathcal{O}(nd + md^2)$	$\mathcal{O}(d^2)$
RESCAL [13]	$\mathcal{O}(nd + md^2)$	$\mathcal{O}(d^2)$
TATEC [64]	$\mathcal{O}(nd + md^2)$	$\mathcal{O}(d^2)$
DistMult [65]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
HolE [62]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d \log d)$
ComplEx [66]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
ANALOGY [68]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
SME (linear) [18]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d^2)$
SME (bilinear) [18]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d^3)$
NTN [19]	$\mathcal{O}(nd + md^2k)$	$\mathcal{O}(d^2k)$
SLM [19]	$\mathcal{O}(nd + mdk)$	$\mathcal{O}(dk)$
MLP [69]	$\mathcal{O}(nd + md)$	$\mathcal{O}(d^2)$
NAM [63]	$\mathcal{O}(nd + md)$	$\mathcal{O}(Ld^2)$

After comparison in complexity, we discuss performance of these models in downstream tasks. Clearly which model is best depends on the task, and also the data. We limit our discussion to the link prediction task (see Section 5.1.1 for details) and WordNet and Freebase data, which is standard practice in previous studies. An interesting observation is that those seemingly more expressive models do not necessarily have better performance. For instance, it was demonstrated that the NTN model performed slightly worse than the simpler MLP model [69], and it even performed worse than TransE and DistMult [65], which are almost the simplest KG embedding models. The reason could be that expressive models often require a large number of parameters and tend to overfit on small- and medium-sized datasets. Nickel et al. [62] recently reimplemented some of the models and compared them in the identical setting, i.e., using the ranking loss of Eq. (2) solved by SGD with AdaGrad. They found that HolE performed substantially better than TransE, TransR, RESCAL, and MLP. Trouillon et al. [66] later showed that ComplEx which defines complex-valued embeddings performed even better than HolE. ANALOGY, which subsumes DistMult, HolE, and ComplEx as special cases, gave the best link prediction results to date reported on WordNet and Freebase data.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

3.5 Other Approaches

Besides the aforementioned models, there are other works which learn representations for head-tail entity pairs rather than individual entities [21]. Specifically, given a triple (h, r, t), the relation r is represented as a vector $\mathbf{r} \in \mathbb{R}^d$, and the entity pair (h, t) another vector $\mathbf{p} \in \mathbb{R}^d$. The plausibility of the fact can be measured by the inner product of $\mathbf{r}$ and $\mathbf{p}$. These vector representations are then learned by minimizing a pairwise ranking loss similar to the one defined in Eq. (2). Such entity pair representation is particularly prevalent in relation extraction, which aims to identify possible relations holding between a pair of entities [88]. Similarly, one can also model the head entity h as a vector $\mathbf{h} \in \mathbb{R}^d$ and the relation-tail entity pair (r, t) as another vector $\mathbf{p} \in \mathbb{R}^d$ [89], [90]. Nevertheless, such paired formulations have their disadvantages. For instance, if the head-tail entity pairs (h_1, t) and (h_2, t) are modeled via different vector representations, the information that they share the same tail entity t will be lost. And also, relations between unpaired entities, e.g., h_3 and t, cannot be effectively discovered. It also leads to an increased space complexity, since a vector representation is computed for each entity pair which requires $\mathcal{O}(n^2 d + md)$ parameters in total.

4 Incorporating Additional Information

The methods introduced so far conduct the embedding task using only facts observed in the KG. In fact, there is a wide variety of additional information that can be incorporated to further improve the task, e.g., entity types, relation paths, textual descriptions, as well as logical rules. In this section, we discuss how such information can be integrated.

4.1 Entity Types

The first kind of additional information we consider is entity types, i.e., semantic categories to which entities belong. For example, `AlfredHitchcock` has the type of `Person`, and `Psycho` the type of `CreativeWork`. This kind of information is available in most KGs, usually encoded by a specific relation and stored also in the form of triples, e.g., (`Psycho`, `IsA`, `CreativeWork`).[9] A straightforward method to model such information, as investigated in [22], is to take `IsA` as an ordinary relation and the corresponding triples as ordinary training examples.

Guo et al. [25] proposed semantically smooth embedding (SSE), which requires entities of the same type to stay close to each other in the embedding space, e.g., `Psycho` is supposed to stay closer to `Avatar` than to `JamesCameron`. SSE employs two manifold learning algorithms, i.e., Laplacian eigenmaps [91] and locally linear embedding [92] to model this smoothness assumption. The former requires an entity to lie close to every other entity in the same category, giving a smoothness measure of

$$\mathcal{R}_1 = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \|\mathbf{e}_i - \mathbf{e}_j\|_2^2 w_{ij}^1,$$

where $\mathbf{e}_i$ and $\mathbf{e}_j$ are the vector embeddings of entities e_i and e_j respectively; $w_{ij}^1 = 1$ if the two entities belong to the same category and $w_{ij}^1 = 0$ otherwise. By minimizing $\mathcal{R}_1$, we expect a small distance between $\mathbf{e}_i$ and $\mathbf{e}_j$ whenever $w_{ij}^1 = 1$. The latter represents an entity as a linear combination of its nearest neighbors, i.e., entities within the same category. The smoothness measure is defined as

$$\mathcal{R}_2 = \sum_{i=1}^{n} \|\mathbf{e}_i - \sum_{e_j \in \mathbb{N}_{e_i}} w_{ij}^2 \mathbf{e}_j\|_2^2,$$

where $\mathbb{N}_{e_i}$ is the set containing nearest neighbors of entity e_i; $w_{ij}^2 = 1$ if $e_j \in \mathbb{N}_{e_i}$ and $w_{ij}^2 = 0$ otherwise.[10] By minimizing $\mathcal{R}_2$, we expect each entity to be linearly reconstructed from its nearest neighbors with low error. $\mathcal{R}_1$ and $\mathcal{R}_2$ are then incorporated as regularization terms to constrain the embedding task. SSE was proved to perform better than the straightforward method in both KG embedding and downstream tasks. A major limitation of SSE is that it assumes entities' semantic categories are non-hierarchical and each entity belongs to exactly one category. This is obviously not the case in typical real-world KGs.

Xie et al. [26] devised type-embodied knowledge representation learning (TKRL), which can handle hierarchical entity categories and multiple category labels. TKRL is a translational distance model with type-specific entity projections. Given a fact (h, r, t), it first projects h and t with type-specific projection matrices, and then models r as a translation between the two projected entities. The scoring function is accordingly defined as

$$f_r(h, t) = -\|\mathbf{M}_{rh}\mathbf{h} + \mathbf{r} - \mathbf{M}_{rt}\mathbf{t}\|_1,$$

where $\mathbf{M}_{rh}$ and $\mathbf{M}_{rt}$ are projection matrices for h and t. To handle multiple category labels, $\mathbf{M}_{rh}$ is represented as a weighted sum of all possible type matrices, i.e.,

$$\mathbf{M}_{rh} = \frac{\sum_{i=1}^{n_h} \alpha_i \mathbf{M}_{c_i}}{\sum_{i=1}^{n_h} \alpha_i}, \quad \alpha_i = \begin{cases} 1, & c_i \in \mathbb{C}_{r_h}, \\ 0, & c_i \notin \mathbb{C}_{r_h}, \end{cases}$$

where $n_h \geq 1$ is the number of categories to which h belongs; c_i the i-th category among them; $\mathbf{M}_{c_i}$ the projection matrix of c_i; α_i the corresponding weight; and $\mathbb{C}_{r_h}$ the set of types that a head entity can have in relation r. To further handle hierarchical categories, $\mathbf{M}_{c_i}$ is represented as a composition of the projection matrices associated with all sub-categories of c_i. Two types of composition operations are used, i.e.,

$$\begin{aligned} \text{addition:} \quad & \mathbf{M}_{c_i} = \beta_1 \mathbf{M}_{c_i^{(1)}} + \cdots + \beta_\ell \mathbf{M}_{c_i^{(\ell)}}; \\ \text{multiplication:} \quad & \mathbf{M}_{c_i} = \mathbf{M}_{c_i^{(1)}} \circ \cdots \circ \mathbf{M}_{c_i^{(\ell)}}. \end{aligned}$$

Here $c_i^{(1)}, \cdots, c_i^{(\ell)}$ are sub-categories of c_i in the hierarchy; $\mathbf{M}_{c_i^{(1)}}, \cdots, \mathbf{M}_{c_i^{(\ell)}}$ their projection matrices; and $\beta_1, \cdots, \beta_\ell$ the corresponding weights. $\mathbf{M}_{rt}$ is defined in a similar way. Although TKRL achieves good performance in downstream tasks such as link prediction and triple classification, it has a relatively high space complexity since it associates each category with a specific projection matrix.

Entity types can also be used as constraints of head and tail positions for different relations, e.g., head entities of relation `DirectorOf` should be those with the type of `Person`,

9. Here we use "`IsA`" to denote the relation that links entities to their semantic categories. Different KGs actually name this relation in their own way, e.g., Freebase adopts `/type/object/type` and NELL uses `Generalization`.

10. w_{ij}^2 are further normalized so that $\sum_{j=1}^{n} w_{ij}^2 = 1$ for each i.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

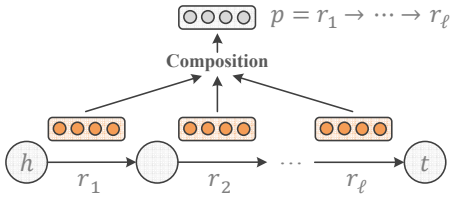


Fig. 4. Path representations are semantic compositions of their relation embeddings. The figure is adapted from [27].

and tail entities those with the type of `CreativeWork`. [77] and [26] tried to impose such constraints in the training process, particularly during the generation of negative training examples. Negative examples that violate entity type constraints are excluded from training [77], or generated with substantially low probabilities [26]. [93] imposed similar constraints on RESCAL, a tensor factorization model. The idea is to discard invalid facts with wrong entity types, and factorize only a sub-tensor composed of remaining facts. See [23], [94] for other approaches in which entity type information is also incorporated.

4.2 Relation Paths

The second kind of additional information we consider is relation paths, i.e., multi-hop relationships between entities.[11] A relation path is typically defined as a sequence of relations $r_1 \to \cdots \to r_\ell$ through which two entities can be connected on the graph. For example, `BornIn` $\to$ `LocatedIn` is a path linking `AlfredHitchcock` to `England`, via an intermediate node `Leytonstone`. Relation paths contain rich semantic cues and are extremely useful for KG completion, e.g., `BornIn` $\to$ `LocatedIn` is indicative of the relation `Nationality` between `AlfredHitchcock` and `England`.

Relation paths have long been studied in multi-relational data. For instance, the path ranking algorithms directly use paths connecting two entities as features to predict potential relations between them [37], [38], [39]. They have, very recently, been integrated into KG embedding. A key challenge then is how to represent such paths in the same vector space along with entities and relations. A straightforward solution is to represent a path as a composition of the representations of its constituent relations, since the semantic meaning of the path depends on all these relations. Fig. 4 provides a simple illustration of this idea.[12] Actually, almost all the existing approaches handle relation paths using composition strategies. Typical composition operations include addition, multiplication, and recurrent neural network (RNN) [95].

Lin et al. [27] proposed an extension of TransE to model relation paths, referred to as path-based TransE (PTransE). Given a path $p = r_1 \to \cdots \to r_\ell$ linking two entities h and t, as well as the vector representations $\mathbf{r}_1, \cdots, \mathbf{r}_\ell$ of the constituent relations, PTransE considers all the three types of composition operations, i.e.,

$$\begin{aligned} \text{addition:}\ & \mathbf{p} = \mathbf{r}_1 + \cdots + \mathbf{r}_\ell; \\ \text{multiplication:}\ & \mathbf{p} = \mathbf{r}_1 \circ \cdots \circ \mathbf{r}_\ell; \\ \text{RNN:}\ & \mathbf{c}_i = f\left(\mathbf{W}[\mathbf{c}_{i-1}; \mathbf{r}_i]\right). \end{aligned}$$

11. Facts themselves are regarded as one-hop relationships.

12. For simplicity relations are represented as vectors in this figure. But they can also be represented in other forms, e.g., matrices [28].

Here, $\mathbf{c}_i$ indicates the accumulated path vector at the i-th relation; $\mathbf{W}$ is a composition matrix shared by all relations; $[\mathbf{c}_{i-1}; \mathbf{r}_i]$ denotes the concatenation of $\mathbf{c}_{i-1}$ and $\mathbf{r}_i$; and f is a non-linearity function. By setting $\mathbf{c}_1 = \mathbf{r}_1$ and recursively traversing from left to right, one can finally get $\mathbf{p} = \mathbf{c}_\ell$. The path p is then required to be consistent with a direct relation r between the two entities, i.e., $\|\mathbf{p} - \mathbf{r}\|_1$ tends to be small if (h, r, t) holds. For each fact $(h, r, t) \in \mathbb{D}^+$, PTransE defines a loss w.r.t. the paths, i.e.,

$$\mathcal{L}_{\text{path}} = \frac{1}{Z} \sum_{p \in \mathbb{P}(h,t)} R(p|h,t) \cdot \ell(p, r),$$

where $\mathbb{P}(h, t)$ is the set of all paths connecting h and t; $R(p|h, t)$ indicates the reliability of a path p given the two entities; $Z = \sum_{p \in \mathbb{P}(h,t)} R(p|h, t)$ is a normalization factor; and $\ell(p, r)$ is a loss specified on the path-relation pair (p, r). The path reliability $R(p|h, t)$ can be calculated by a network-based resource-allocation mechanism [96], and the loss $\ell(p, r)$ is defined as

$$\ell(p, r) = \sum_{r'} \max(0, \gamma + \|\mathbf{p} - \mathbf{r}\|_1 - \|\mathbf{p} - \mathbf{r}'\|_1),$$

which prefers a lower value of $\|\mathbf{p}-\mathbf{r}\|_1$ than of any $\|\mathbf{p}-\mathbf{r}'\|_1$. Finally, to learn entity and relation representations, $\mathcal{L}_{\text{path}}$ is aggregated over all the facts in $\mathbb{D}^+$ and then combined with the original TransE loss. Experimental results showed that by further incorporating relation paths, PTransE can perform substantially better than TransE in KG completion and relation extraction.

Guu et al. [28] proposed a similar framework, the idea of which is to build triples using entity pairs connected not only with relations but also with relation paths. For example, given a pair of entities (h, t) and a path $p = r_1 \to \cdots \to r_\ell$ between them, a new triple (h, p, t) can be constructed. To model such path-connected triples, Guu et al. devised extensions of both the TransE model and the RESCAL model. The former uses the addition composition, and defines the score of (h, p, t) as

$$f_p(h, t) = -\|\mathbf{h} + (\mathbf{r}_1 + \cdots + \mathbf{r}_\ell) - \mathbf{t}\|_1,$$

while the latter chooses the multiplication composition and defines the score as

$$f_p(h, t) = \mathbf{h}^\top (\mathbf{M}_1 \circ \cdots \circ \mathbf{M}_\ell)\mathbf{t}.$$

Path-connected triples are then treated the same as those relation-connected ones during training. This approach was shown to perform well in answering path queries on KGs. A more limited version of the same approach was simultaneously introduced in [97].

While incorporating relation paths improves model performance, the huge number of paths poses a critical complexity challenge. Both [27] and [28] had to make approximations by sampling or pruning. To enable efficient path modeling, [29] proposed a dynamic programming algorithm which can incorporate all relation paths of bounded length. Moreover, in this work, not only relations but also intermediate nodes (i.e. entities) are modeled in compositional path representations. For other related work, please refer to [98], [99], [100].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

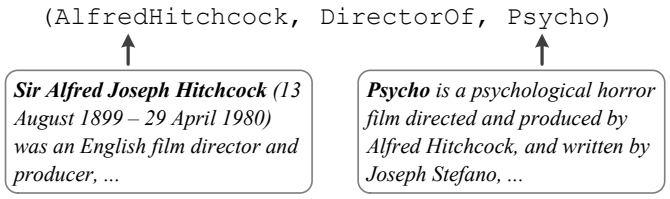


Fig. 5. Examples of entity descriptions. The figure is adapted from [32].

4.3 Textual Descriptions

This section discusses the incorporation of textual descriptions for entities. Actually, in most KGs there are concise descriptions for entities which contain rich semantic information about them. Fig. 5 shows the descriptions for `AlfredHitchcock` and `Psycho` in Freebase. Besides entity descriptions stored in KGs, it can be extended to incorporate more general textual information such as news releases [19] and Wikipedia articles [30], [31], [33].

Embedding KGs with textual information dates back to the NTN model [19], where textual information is simply used to initialize entity representations. Specifically, NTN first learns word vectors from an auxiliary news corpus, and then initializes the representation of each entity by averaging the vectors of words contained in its name. For example, the embedding of `AlfredHitchcock` is initialized by the average word vectors of "*alfred*" and "*hitchcock*". A similar method was later proposed in [76], which represents entities as average word vectors of their descriptions rather than just their names. This kind of methods model textual information separately from KG facts, and hence fail to leverage interactions between them.

Wang et al. [30] proposed the first joint model which can make better use of textual information during embedding. The key idea is to align the given KG with an auxiliary text corpus, and then jointly conduct KG embedding and word embedding. As such, entities/relations and words are represented in the same vector space and operations like inner product (similarity) between them are hence meaningful. The joint model has three components: knowledge model, text model, and alignment model. The knowledge model is to embed entities and relations in the KG. It is a variant of TransE, with a loss $\mathcal{L}_K$ to measure the fitness to KG facts. The text model is to embed words in the text corpus. It is a variant of Skip-gram [101], with a loss $\mathcal{L}_T$ to measure the fitness to co-occurring word pairs. Finally, the alignment model guarantees the embeddings of entities/relations and words lie in the same space. Different mechanisms of alignment are introduced, e.g., alignment by entity names [30], by Wikipedia anchors [30], and by entity descriptions [31]. A loss $\mathcal{L}_A$ is defined to measure the quality of alignment. The joint model then amounts to minimizing a loss aggregated from the three components, i.e.,

$$\mathcal{L} = \mathcal{L}_K + \mathcal{L}_T + \mathcal{L}_A.$$

Jointly embedding utilizes information from both structured KGs and unstructured text. KG embedding and word embedding can thus be enhanced by each other. Moreover, by aligning these two types of information, jointly embedding enables the prediction of out-of-KG entities, i.e., phrases appearing in web text but not included in the KG yet.

Xie et al. [32] proposed description-embodied knowledge representation learning (DKRL), the aim of which is to extend TransE so as to further handle entity descriptions. DKRL associates each entity e with two vector representations, i.e., a structure-based $\mathbf{e}_s$ and a description-based $\mathbf{e}_d$. The former captures structural information conveyed in KG facts, while the latter captures textual information expressed in the entity description. The description-based representation is constructed by the constituent word embeddings, via either a continuous bag-of-words encoder or a convolutional neural network encoder. Given a fact (h, r, t), DKRL defines the scoring function as

$$\begin{aligned} f_r(h,t) = &- \|\mathbf{h}_s + \mathbf{r} - \mathbf{t}_s\|_1 - \|\mathbf{h}_d + \mathbf{r} - \mathbf{t}_d\|_1 \\ &- \|\mathbf{h}_s + \mathbf{r} - \mathbf{t}_d\|_1 - \|\mathbf{h}_d + \mathbf{r} - \mathbf{t}_s\|_1, \end{aligned}$$

where $\mathbf{r}$ is the vector representation of the relation, shared by both structure-based $\mathbf{h}_s/\mathbf{t}_s$ and description-based $\mathbf{h}_d/\mathbf{t}_d$. Entity, relation, and word embeddings can then be learned simultaneously by minimizing the ranking loss defined in Eq. (2). Experimental results demonstrated the superiority of DKRL over TransE, particularly in the zero-shot scenario with out-of-KG entities.

Wang et al. [33] recently proposed a text-enhanced KG embedding model, referred to as TEKE. Given a KG and a text corpus, TEKE first annotates entities in the corpus and constructs a co-occurrence network composed of entities and words. Then, for each entity e, TEKE defines its textual context $n(e)$ as its neighbors in the co-occurrence network, i.e., words co-occurring frequently with the entity in the text corpus. A textual context embedding $\mathbf{n}(e)$ is further introduced for that entity, defined as the weighted average of the word vectors in $n(e)$.[13] For each relation r in a fact (h, r, t), TEKE defines its textual context as the common neighbors of h and t, i.e., $n(h, t) = n(h) \cap n(t)$. A textual context embedding $\mathbf{n}(h, t)$ is similarly defined for that relation. Textual context embeddings are then incorporated into traditional methods, e.g., TransE, to learn more expressive entity and relation representations such as

$$\begin{aligned} \hat{\mathbf{h}} &= \mathbf{A}\mathbf{n}(h) + \mathbf{h}, \\ \hat{\mathbf{t}} &= \mathbf{A}\mathbf{n}(t) + \mathbf{t}, \\ \hat{\mathbf{r}} &= \mathbf{B}\mathbf{n}(h, t) + \mathbf{r}. \end{aligned}$$

Here, $\mathbf{A}$, $\mathbf{B}$ are weight matrices, and $\mathbf{h}$, $\mathbf{t}$, $\mathbf{r}$ bias vectors. This extension also applies for TransH and TransR. By incorporating textual context embeddings, TEKE was proved to outperform the original models of TransE, TransH, and TransR. See [102], [103], [104], [105] for other approaches where textual information is also considered.

4.4 Logical Rules

Finally we consider the incorporation of logical rules, particularly those represented in terms of first-order Horn clauses, e.g., $\forall x, y : \texttt{HasWife}(x, y) \Rightarrow \texttt{HasSpouse}(x, y)$ stating that any two entities linked by the relation `HasWife` should also be linked by the relation `HasSpouse`. Such logical rules contain rich background information and have been widely studied in knowledge acquisition and inference, usually on

13. Word vectors are pre-trained using the word2vec tool [101]. They are not learned jointly with entity and relation embeddings.

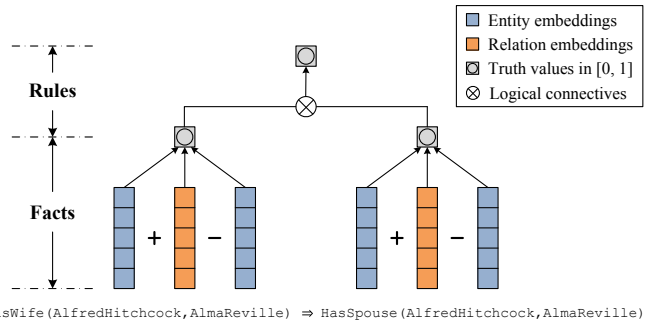


Fig. 6. Simple illustration of KALE. The figure is adapted from [34].

the basis of Markov logic networks [40], [41], [42]. There are also systems such as WARMR [106], ALEPH [107], and AMIE [108], [109] which can extract logical rules automatically from KGs. Recently, there has been growing interest in combining logical rules with KG embedding models.

Wang et al. [23] tried to utilize rules to refine embedding models during KG completion. In their work, KG completion is formulated as an integer linear programming problem, with the objective function generated from embedding models and the constraints from rules. Facts inferred in this way will be the most preferred by the embedding models and comply with all the rules. A similar work that combines rules and embedding models via Markov logic networks was later introduced in [24]. However, in [23] and [24], rules are modeled separately from embedding models, serving as post-processing steps, and thus will not help to obtain better embeddings.

Guo et al. [34] proposed a joint model which embeds KG facts and logical rules simultaneously. A key ingredient of this approach, called KALE, is to represent and model facts and rules in a unified framework. Specifically, a fact (h, r, t) is taken as a ground atom, with its truth value defined as

$$I(h, r, t) = 1 - \frac{1}{3\sqrt{d}}\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_1,$$

where d is the dimension of the embeddings, and $I(h, r, t) \in [0, 1]$ a linearly transformed version of the TransE score indicating how likely the fact holds. Logical rules are first instantiated into ground rules, e.g., the universally quantified rule $\forall x, y : \texttt{HasWife}(x, y) \Rightarrow \texttt{HasSpouse}(x, y)$ can be grounded into $\texttt{HasWife}(\texttt{AlfredHitchcock}, \texttt{AlmaReville}) \Rightarrow \texttt{HasSpouse}(\texttt{AlfredHitchcock}, \texttt{AlmaReville})$. Ground rules are then interpreted as complex formulae constructed by combining ground atoms with logical connectives (e.g. $\wedge$ and $\Rightarrow$), and modeled by t-norm fuzzy logics [110]. The truth value of a ground rule is a composition of the truth values of the constituent ground atoms, via specific t-norm based logical connectives, e.g.,

$$I(f_1 \Rightarrow f_2) = I(f_1) \cdot I(f_2) - I(f_1) + 1,$$
$$I(f_1 \wedge f_2 \Rightarrow f_3) = I(f_1) \cdot I(f_2) \cdot I(f_3) - I(f_1) \cdot I(f_2) + 1.$$

This value also lies within the range of $[0, 1]$, indicating to what degree the ground rule is satisfied. In this way, KALE represents facts and rules in a unified framework, as atomic and complex formulae respectively. Fig. 6 provides a simple illustration of this framework. After unifying facts and rules, KALE minimizes a global loss involving both of them to learn entity and relation embeddings. The learned embeddings are hence compatible not only with facts but also with rules, which can be more effective for knowledge acquisition and inference.

Rocktäschel et al. [35] devised a model similar to KALE. In their work, however, vector embeddings are introduced for entity pairs rather than individual entities, making it particularly useful for relation extraction. Since entities do not have their own embeddings, relations between unpaired entities cannot be effectively discovered. Both [34] and [35] share a common drawback: they have to instantiate universally quantified rules into ground rules before learning their models. This grounding procedure could be extremely time and space inefficient especially when there are a huge number of entities in the KG and/or the rules are complicated themselves. To address this drawback, Demeester et al. [111] recently proposed an extension of [35], the key idea of which is to impose logical implications by regularizing relation embeddings so as to avoid grounding. For example, given a universally quantified rule $\forall x, y : \texttt{HasWife}(x, y) \Rightarrow \texttt{HasSpouse}(x, y)$, [111] tried to model it by using only the embeddings of the two relations `HasWife` and `HasSpouse`, without instantiating x and y with concrete entities in the KG. Nevertheless, this strategy works only for rules in the simplest form of $\forall x, y : r_i(x, y) \Rightarrow r_j(x, y)$, and cannot be generalized to more complicated rules. For other related work, please refer to [112], [113].

4.5 Other Information

Besides the four types of additional information mentioned above, there are also a few studies which tried to incorporate other information into KG embedding.

Entity Attributes. Nickel et al. [22] argued that relations in KGs can indicate both relationships between entities (e.g. (`AlfredHitchcock`, `HasWife`, `AlmaReville`)) and attributes of entities (e.g. (`AlfredHitchcock`, `Gender`, `Male`)), but most KG embedding techniques do not explicitly distinguish between relations and attributes. Take the tensor factorization model RESCAL as an example. In this model, each KG relation is encoded as a slice of the tensor, no matter it indicates a true relation or just an attribute. This naive handling of attributes will dramatically increase the dimensionality of the tensor, while a huge amount of entries in this tensor, however, will be wasted.[14] To address this problem, [22] proposed to explicitly distinguish attributes from relations. Relations are still encoded in a tensor, while attributes in a separate entity-attribute matrix. This matrix is then factorized jointly with the tensor to learn representations simultaneously for entities, relations, and attributes. A similar idea was later studied in a translational distance model [114].

Temporal Information. Jiang et al. [115] observed that KG facts are usually time-sensitive, e.g., (`AlfredHitchcock`, `BornIn`, `Leytonstone`) happened in the year of 1899, while (`AlfredHitchcock`, `DiedIn`, `BelAir`) took place in

14. Note that as opposed to true entities (e.g. `AlfredHitchcock` and `AlmaReville`), attribute values (e.g. `Male`) might never occur as head entities in a relation or even as tail entities in other relations except for the specific attribute.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

the year of 1980. Based on this observation they proposed a time-aware embedding model. The idea of this model is to impose temporal order constraints on time-sensitive relation pairs, e.g., `BornIn` and `DiedIn`. Given such a pair (r_i, r_j), the prior relation is supposed to lie close to the subsequent relation after a temporal transition, i.e., $\mathbf{M}\mathbf{r}_i \approx \mathbf{r}_j$, where $\mathbf{M}$ is a transition matrix capturing the temporal order information between relations. After imposing such temporal order constraints, [115] is able to learn temporally consistent relation embeddings. Esteban et al. [116] tried to model the temporal evolution of KGs. In their model, changes in a KG always arrive as events, represented by labeled quadruples such as $(h, r, t, s; \text{True})$ or $(h, r, t, s; \text{False})$, indicating that the fact (h, r, t) appears or vanishes at time s, respectively. Each quadruple is then modeled as a four-way interaction among $\mathbf{h}$, $\mathbf{r}$, $\mathbf{t}$, and $\mathbf{s}$, where $\mathbf{s}$ is the vector representation of the time stamp. This model was shown to perform well in dynamic domains, e.g., medical and sensor data. Trivedi et al. [117] recently proposed to learn non-linearly evolving entity representations over time so as to perform temporal reasoning over dynamic KGs. Each fact in a dynamic KG is represented as a quadruple (h, r, t, s), indicating the creation of relation r between head entity h and tail entity t at time s. A temporal point process [118] is then employed to model occurrence of facts, with a bilinear scoring function to capture multi-relational interactions between entities, and a deep recurrent neural network to learn non-linearly evolving entity representations. This approach performed quite well in link prediction, in particular, time prediction.

Graph Structures. Feng et al. [119] proposed a graph-aware embedding model which learns entity and relation representations by leveraging three types of graph structures. The first is neighbor context, which is actually equivalent to triples observed in a KG. The second is path context, very similar to relation paths discussed in Section 4.2.[15] The last is edge context, which has not been considered in previously introduced methods. Given a specific entity, its edge context is defined as all kinds of relations linking to and from that entity, based simply on the intuition that all these relations are representative of that entity. For example, the edge context of `AlfredHitchcock` might include relations such as `BornIn`, `DiedIn`, `HasWife`, and `DirectorOf`, all indicating `AlfredHitchcock` to be a `Person` or more specifically, a `Director`. Experimental results further demonstrated the effectiveness of modeling these graph structures. Jiang et al. [121] suggested to estimate the plausibility of a fact from its immediate context. The immediate context of a fact (h, r, t) is defined as: (i) triples where h is the head, (ii) triples where h is the tail, (iii) triples where t is the head, (iv) triples where t is the tail, and (v) triples with arbitrary relations but where the two entities are h and t. This work was shown effective in predicting links on multi-relational data.

Evidence from Other Relational Learning Methods. There is another line of research which combines KG embedding with other relational learning methods, e.g., the path ranking algorithm (PRA), to take the strengths of different types of methods. Dong et al. [69] proposed to combine MLP with PRA via a fusion system. Specifically, after fitting the two models separately, they used the outputs of MLP and PRA as scalar features, and learned a final fusion layer by training a binary classifier. They found that fusing these two models improves performance: the fused system obtained a result of 0.911 for the area under the ROC curve, as compared to 0.882 for MLP and 0.884 for PRA on their specific dataset. Nickel et al. [122] designed a generic framework to combine latent and observable variable models. In particular, if it combines RESCAL with PRA, the scoring function becomes $f_r(h, t) = \mathbf{h}^\top \mathbf{M}_r \mathbf{t} + \mathbf{w}_r^\top \boldsymbol{\phi}_{ht}$. The first term is the RESCAL score and the second term the PRA score, in which $\boldsymbol{\phi}_{ht}$ is a feature vector composed of path features and $\mathbf{w}_r$ holds the weights of these features. This is a joint model which can be trained by alternately optimizing the RESCAL parameters with the PRA parameters. After the combination, RESCAL only needs to model the "residual errors" that cannot be modeled by PRA, which requires lower latent dimensionality and speeds up training. For more details about these combined models, refer to [36] and references therein.

15. The difference lies in that Section 4.2 considers only relations in a path, while path context further takes into account intermediate nodes in that path, as practiced in [29], [100], [120].

5 Applications in Downstream Tasks

After a systematic review of currently available KG embedding techniques, this section explores how the learned entity and relation embeddings can be applied to and benefit a wide variety of downstream tasks. We categorize such tasks into (i) in-KG applications and (ii) out-of-KG applications, discussed as follows.

5.1 In-KG Applications

In-KG applications are those conducted within the scope of the KG where entity and relation embeddings are learned. We introduce four such applications, i.e., link prediction, triple classification, entity classification, and entity resolution, which have been extensively studied in the literature. All these applications are sorts of refinement (e.g. completion or de-duplication) of the input KG [123], from different viewpoints and application context.

5.1.1 Link Prediction

Link prediction is typically referred to as the task of predicting an entity that has a specific relation with another given entity, i.e., predicting h given (r, t) or t given (h, r), with the former denoted as $(?, r, t)$ and the latter as $(h, r, ?)$. For example, (`?`, `DirectorOf`, `Psycho`) is to predict the director of the film, while (`AlfredHitchcock`, `DirectorOf`, `?`) amounts to predicting films directed by that specific person. This is essentially a KG completion task, i.e., adding missing knowledge to the graph, and has been tested extensively in previous literature [14], [15], [16], [62]. This link prediction task is also sometimes called entity prediction [27] or entity ranking [18]. A similar idea can also be used to predict relations between two given entities, i.e., $(h, ?, t)$, which is usually referred to as relation prediction [26], [27].

With entity and relation representations learned beforehand, link prediction can be carried out simply by a ranking procedure. Take the prediction task $(?, r, t)$ as an example. To predict the head, one can take every entity h' in the KG as a candidate answer and calculate a score $f_r(h', t)$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

for each (h', r, t). This can easily be achieved by using the learned embeddings and scoring function once an embedding model has been trained on the KG, e.g., $f_r(h', t) = -\|\mathbf{h}'+\mathbf{r}-\mathbf{t}\|_{1/2}$ if TransE has been employed for KG embedding. Ranking these scores in descending order will result in a ranked list of candidate answers. For instance, given the prediction task (?, DirectorOf, Psycho), one may generate an ordered list {JamesCameron, AlfredHitchcock, GeorgeLucas, QuentinTarantino} by using this ranking procedure. The prediction task $(h, r, ?)$ or $(h, ?, t)$ can be carried out in a similar manner.

For evaluation, a common practice is to record ranks of correct answers in such ordered lists, so as to see whether correct answers can be ranked before incorrect ones. In the forementioned example of (?, DirectorOf, Psycho), the correct answer AlfredHitchcock gets a rank of 2. Lower ranks indicate better performance. Various evaluation metrics have been designed based on such ranks, e.g., mean rank (the average of predicted ranks), mean reciprocal rank (the average of reciprocal ranks), Hits@n (the proportion of ranks no larger than n), and AUC-PR (the area under the precision-recall curve).

5.1.2 Triple Classification

Triple classification consists in verifying whether an unseen triple fact (h, r, t) is true or not, e.g., (AlfredHitchcock, DirectorOf, Psycho) should be classified as a true fact while (JamesCameron, DirectorOf, Psycho) a false one. This task, again, can be seen as some sort of completion of the input KG, which has also been studied extensively in previous works [15], [16], [19].

Recall that once an embedding model has been learned on the KG, we can calculate a score for any triple (h, r, t) as long as $h, t \in \mathbb{E}$ and $r \in \mathbb{R}$, e.g., $f_r(h, t) = -\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|_{1/2}$ if the TransE model has been learned. Triple classification can then be carried out simply on the basis of such triple scores. Triples with higher scores tend to be true facts. Specifically, we introduce for each relation r a threshold δ_r. Then any unseen fact from that relation, say (h, r, t), will be predicted as true if its score $f_r(h, t)$ is higher than δ_r, and as false otherwise.[16] In this way, we obtain a triple classifier for each relation. Traditional metrics for classification can be used to evaluate this task, e.g., micro- and macro-averaged accuracy [25]. Since for each triple a real valued score will be output along with the binary label, ranking metrics can also be used here, e.g., mean average precision [34].

5.1.3 Entity Classification

Entity classification aims to categorize entities into different semantic categories, e.g., AlfredHitchcock is a Person, and Psycho a CreativeWork. Given that in most cases the relation encoding entity types (denoted as IsA) is contained in the KG and has already been included into the embedding process, entity classification can be treated as a specific link prediction task, i.e., $(x, \text{IsA}, ?)$. Similar prediction and evaluation procedures can be applied here (see Section 5.1.1 for details). Entity classification is obviously a KG completion problem, and has been studied in [13] and [22].

16. The relation-specific threshold δ_r can be determined by using a small set of facts observed for that relation, either a subset of $\mathbb{D}^+$ or a separate development set.

5.1.4 Entity Resolution

Entity resolution consists in verifying whether two entities refer to the same object. In some KGs many nodes actually refer to identical objects, e.g., in the Cora dataset [124] which contains citations with the fields of author, title, and venue, the name of an author or a venue can be written in different ways. Entity resolution is the task that de-duplicates such nodes [13], [18].

Bordes et al. [18] considered a scenario where the KG already contains a relation stating whether two entities are equivalent (denoted as EqualTo) and an embedding has been learned for that relation. In this case, entity resolution degenerates to a triple classification problem, i.e., to judge whether the triple $(x, \text{EqualTo}, y)$ holds or how likely this triple holds. Triple scores output by an embedding model can be directly used for such prediction (see Section 5.1.2 for details). This intuitive strategy, however, does not always work since not all KGs encode the EqualTo relation. Nickel et al. [13] proposed to perform entity resolution solely on the basis of entity representations. More specifically, given two entities x, y and their vector representations $\mathbf{x}$, $\mathbf{y}$, the similarity between x and y is computed as $k(x, y) = e^{-\|\mathbf{x}-\mathbf{y}\|_2^2/\sigma}$, and this similarity score is used to measure the likelihood that x and y refer to the same entity. The new strategy works even if the EqualTo relation is not encoded in the input KG. AUC-PR is the most widely adopted evaluation metric for this task.

5.2 Out-of-KG Applications

Out-of-KG applications are those which break through the boundary of the input KG and scale to broader domains. We introduce three such applications as examples, including relation extraction, question answering, and recommender systems. We do not seek to provide systematic reviews of these tasks or introduce the state-of-the-arts. But instead we focus particularly on showing how KG embedding can be applied to these domains. And we hope they can provide new insights into future application of KG embedding.

5.2.1 Relation Extraction

Relation extraction aims to extract relational facts from plain text where entities have already been detected. For example, given a sentence "*Alfred Hitchcock directed Psycho*" with the entities $h =$ AlfredHitchcock and $t =$ Psycho detected, a relation extractor should predict the relation DirectorOf between these two entities. Relation extraction has long been a crucial task in natural language processing, and provides an effective means to enrich KGs. Much research has tried to leverage KGs for this task, but usually as distant supervision to automatically generate labeled data [9], [125], [126], [127]. Such approaches are still text-based extractors, ignoring the capability of a KG itself to reason new facts.

Recently, Weston et al. [20] proposed to combine TransE with a text-based extractor so as to better perform relation extraction. Specifically, in the training phase, they learned a text-based extractor from a text corpus and a TransE model from a KG aligned to that corpus. The text-based extractor scores the similarity between each relation r and its textual mention m, i.e., $S_{\text{text}}(m, r)$. These scores can then be used to predict relations from their textual mentions, i.e., evidence

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

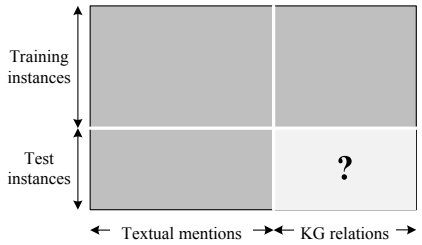


Fig. 7. A matrix encoding text and KGs. The figure is adapted from [21].

from the text corpus. Meanwhile, the TransE model scores the plausibility of each missing fact (h, r, t) in the KG, i.e., $S_{\text{KG}}(h, r, t)$.[17] These scores can be used to predict relations from their interactions with entities in the KG, i.e., evidence from the KG. In the test phase, given two entities h, t and all relation mentions $\mathbb{M}_{h,t}$ associated with them, a prediction $\hat{r}$ is first made with the text-based extractor, and a composite score is then introduced for the candidate fact,[18] i.e.,

$$S_{\text{text+KG}}(h, \hat{r}, t) = \sum_{m \in \mathbb{M}_{h,t}} S_{\text{text}}(m, \hat{r}) + S_{\text{KG}}(h, \hat{r}, t).$$

This composite model favors predictions that agree with not only the textual mentions but also the KG. Experimental results further showed that incorporating the TransE model can successfully improve over traditional text-based extractors. Similar improvements have also been observed after incorporating TransH [15] and TransR [16].

Riedel et al. [21] devised a different framework which performs relation extraction by jointly embedding plain text and KGs. In their work, text and KGs are represented in the same matrix. Each row of the matrix stands for a pair of entities, and each column a textual mention or a KG relation. If two entities co-occur with a mention in plain text or with a relation in KGs, the corresponding entry is set to one, and otherwise to zero. For training instances (entity pairs), one can observe both textual mentions and KG relations, with the latter as distant supervision. But for test instances, only textual mentions are available. Relation extraction then is to predict missing KG relations for test instances. Fig. 7 gives a simple illustration of this scenario. Collaborative filtering techniques are further employed for this task, which factorize the input matrix to learn vector embeddings for entity pairs, textual mentions, and KG relations. This framework also improves over traditional text-based extractors. Fan et al. [88] proposed a similar approach to relation extraction. But in their work, the first group of columns in the matrix correspond to text features rather than textual mentions, and matrix completion techniques [128] are employed instead of matrix factorization ones. Chang et al. [93] later devised a tensor-based variant, which encodes plain text and KGs in a three-mode tensor and then factorizes the tensor by using the RESCAL model.

5.2.2 Question Answering

This article considers a specific question answering task, i.e., question answering over KGs. Given a question expressed in natural language, the task is to retrieve the correct answer supported by a triple or set of triples from a KG [11], [12]. Here we show some examples of questions, answers, and supporting triple(s):

- *Who directed Psycho?* – `AlfredHitchcock` (`AlfredHitchcock`, `DirectorOf`, `Psycho`)
- *Where was A. Hitchcock born?* – `Leytonstone` (`AlfredHitchcock`, `BornIn`, `Leytonstone`)
- *What was the nationality of A. Hitchcock?* – `England` (`AlfredHitchcock`, `BornIn`, `Leytonstone`) (`Leytonstone`, `LocatedIn`, `England`)

The use of KGs simplifies question answering by organizing a great variety of answers in a structured format. However, it remains a challenging task because of the great variability of natural language and of the large scale of KGs.

Bordes et al. [11], [12] introduced an embedding-based framework for this task. The key idea of their approach is to learn low-dimensional vector embeddings of words and of KG constituents, so that representations of questions and of their corresponding answers are close to each other in the embedding space. Specifically, let q denote a question and a a candidate answer. A function $S(q, a)$, based on vector embeddings, is designed to score the similarity between the question and the answer, i.e.,

$$S(q, a) = \big(\mathbf{W}\boldsymbol{\phi}(q)\big)^{\top}\big(\mathbf{W}\boldsymbol{\psi}(a)\big).$$

Here $\mathbf{W}$ is a matrix containing embeddings of words, entities, and relations; $\boldsymbol{\phi}(q)$ and $\boldsymbol{\psi}(a)$ are two sparse vectors, the former indicating occurrences of words in the question, and the latter occurrences of entities and relations in the answer.[19] $\mathbf{W}\boldsymbol{\phi}(q)$ and $\mathbf{W}\boldsymbol{\psi}(a)$ are vector representations of the question and answer respectively in the embedding space. Both are combinations of embeddings of their constituents, i.e., words, entities, and relations. $S(\cdot, \cdot)$ generates a high score if a is the correct answer to the question q, and a low score otherwise. Given a training set consisting of questions paired with their correct answers, the embeddings $\mathbf{W}$ can be learned by using typical pairwise ranking optimization, which enforces the score of a correct pair to be higher than that of any incorrect one. The training set can be created by crowdsourcing [129] or by automatically generalizing seed patterns over KGs [130]. Once $\mathbf{W}$ is trained, at test time, for a given question q the answer is predicted as

$$\hat{a} = \text{argmax}_{a \in \mathbb{A}(q)} S(q, a),$$

where $\mathbb{A}(q)$ is the candidate answer set. Bordes et al. empirically demonstrated that this intuitive approach achieves promising results without using any lexicon, rules or additional steps for part-of-speech tagging, syntactic or dependency parsing during training as most traditional question answering systems do.

5.2.3 Recommender Systems

Recommender systems provide advice to users about items they may wish to purchase or examine. Among different recommendation strategies, collaborative filtering techniques

17. $S_{\text{KG}}(h, r, t)$ is a variant of the TransE score $f_r(h, t)$.

18. If the prediction $\hat{r}$ is NA, i.e., a marker indicating that there is no relation between h and t, $S_{\text{KG}}(h, \hat{r}, t)$ will not be introduced.

19. The answer can be a single entity, a triple, a relation path, or even a subgraph [12]. If, for example, it is a single entity, $\boldsymbol{\psi}(a)$ is a one-hot vector with 1 corresponding to the entity, and 0 elsewhere.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

which model the interaction between a user and an item as a product of their latent representations, have achieved significant success [131]. Such techniques, however, do not always work well, since user-item interactions can be very sparse. In this case, hybrid recommender systems, which combine user-item interactions and auxiliary information of users or items, can usually achieve better performance [132].

Zhang et al. [133] recently proposed a hybrid recommendation framework which leverages heterogeneous information in a KG to improve the quality of collaborative filtering. Specifically, they used three types of information stored in the KG, including structural knowledge (triple facts), textual knowledge (e.g. the textual summary of a book or a movie), and visual knowledge (e.g. a book's front cover or a movie's poster image), to derive semantic representations for items. To model the structural knowledge, a typical KG embedding technique, i.e., TransR, is applied which learns a structural representation for each item. For the other two types of information, stacked de-noising auto-encoders and stacked convolutional auto-encoders are employed to extract items' textual representations and visual representations, respectively. Then, to conduct collaborative filtering, each user i is represented as a latent vector $\mathbf{u}_i$, and each item j a latent vector

$$\mathbf{e}_j = \mathbf{s}_j + \mathbf{t}_j + \mathbf{v}_j + \boldsymbol{\eta}_j,$$

where $\mathbf{s}_j$, $\mathbf{t}_j$, and $\mathbf{v}_j$ are the structural, textual, and visual representations associated to that item respectively, and $\boldsymbol{\eta}_j$ is an offset vector. The preference of user i for item j is then modeled as a product of the two latent vectors, i.e., $\mathbf{u}_i^\top \mathbf{e}_j$. Ranking optimization over pair-wise preference is used to learn these latent vectors. Finally, at test time, given a target user i, item recommendation can be made according to the following ranking criterion:

$$i : j_1 \succ j_2 \succ \cdots \succ j_n \;\Leftrightarrow\; \mathbf{u}_i^\top \mathbf{e}_{j_1} > \mathbf{u}_i^\top \mathbf{e}_{j_2} > \cdots > \mathbf{u}_i^\top \mathbf{e}_{j_n},$$

where $i : j_s \succ j_t$ means that user i prefers item j_s over j_t. Experimental results demonstrated the effectiveness of the three types of item representations learned from the KG in recommender systems.

6 Concluding Remarks

KG embedding, which aims to embed entities and relations into continuous vector spaces, has found important applications in various entity-oriented tasks and quickly gained massive attention. This article provided a systematic review of currently available techniques, particularly based on the type of information used in KG embedding. State-of-the-art techniques which conduct embedding using only facts observed in a given KG were first introduced. We described the overall framework, specific model design, typical training procedures, as well as pros and cons of such techniques. After that, some more advanced techniques which perform KG embedding with other information besides facts were later discussed. We focused specifically on the incorporation of four types of additional information, i.e., entity types, relation paths, textual descriptions, and logical rules. The investigation on incorporating additional information has just started, and might receive increasing attention in the near future. Finally, this article explored the application of KG embedding. Two types of applications were introduced, i.e., in-KG applications conducted within the scope of the input KG and out-of-KG applications that scale to broader domains. We hope this brief exploration can provide new insights into future application of KG embedding.

Acknowledgments

We would like to thank all the reviewers for their insightful and valuable suggestions, which significantly improve the quality of this survey. This work is supported by the National Key Research and Development Program of China (grants No. 2016QY03D0503 and 2016QY03D0505) and the National Natural Science Foundation of China (grants No. 61402465 and 61502477). The first author is supported by the Institute of Information Engineering, Chinese Academy of Sciences and the State Key Laboratory of Information Security, Chinese Academy of Sciences (grant No. Y7Z0261101).

References

[1] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proc. of ACM SIGMOD Int. Conf. on Manage. Data*, 2008, pp. 1247–1250.

[2] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer *et al.*, "DBpedia: A large-scale, multilingual knowledge base extracted from Wikipedia," *Semantic Web J.*, vol. 6, no. 2, pp. 167–195, 2015.

[3] F. M. Suchanek, G. Kasneci, and G. Weikum, "YAGO: A core of semantic knowledge," in *Proc. 16th Int. Conf. on World Wide Web*, 2007, pp. 697–706.

[4] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr, and T. M. Mitchell, "Toward an architecture for never-ending language learning," in *Proc. 24th AAAI Conf. Artif. Intell.*, 2010, pp. 1306–1313.

[5] J. Berant, A. Chou, R. Frostig, and P. Liang, "Semantic parsing on Freebase from question-answer pairs," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2013, pp. 1533–1544.

[6] L. P. Heck, D. Hakkani-Tür, and G. Tür, "Leveraging knowledge graphs for web-scale unsupervised semantic parsing," in *INTERSPEECH*, 2013, pp. 1594–1598.

[7] D. Damljanovic and K. Bontcheva, "Named entity disambiguation using linked data," in *Proc. 9th Extended Semantic Web Conf.*, 2012.

[8] Z. Zheng, X. Si, F. Li, E. Y. Chang, and X. Zhu, "Entity disambiguation with Freebase," in *Proc. IEEE/WIC/ACM Int. Joint Conf. Web Intell. Intell. Agent Technol.*, 2012, pp. 82–89.

[9] R. Hoffmann, C. Zhang, X. Ling, L. Zettlemoyer, and D. S. Weld, "Knowledge-based weak supervision for information extraction of overlapping relations," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics*, 2011, pp. 541–550.

[10] J. Daiber, M. Jakob, C. Hokamp, and P. N. Mendes, "Improving efficiency and accuracy in multilingual entity extraction," in *Proc. 9th Int. Conf. Semantic Syst.*, 2013, pp. 121–124.

[11] A. Bordes, J. Weston, and N. Usunier, "Open question answering with weakly supervised embedding models," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2014, pp. 165–180.

[12] A. Bordes, S. Chopra, and J. Weston, "Question answering with subgraph embeddings," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2014, pp. 615–620.

[13] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 809–816.

[14] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Adv. Neural Inf. Process. Syst.*, 2013, pp. 2787–2795.

[15] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 1112–1119.

[16] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2181–2187.

[17] R. Jenatton, N. L. Roux, A. Bordes, and G. R. Obozinski, "A latent factor model for highly multi-relational data," in *Adv. Neural Inf. Process. Syst.*, 2012, pp. 3167–3175.

[18] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "A semantic matching energy function for learning with multi-relational data," *Mach. Learn.*, vol. 94, no. 2, pp. 233–259, 2014.

[19] R. Socher, D. Chen, C. D. Manning, and A. Y. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Adv. Neural Inf. Process. Syst.*, 2013, pp. 926–934.

[20] J. Weston, A. Bordes, O. Yakhnenko, and N. Usunier, "Connecting language and knowledge bases with embedding models for relation extraction," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2013, pp. 1366–1371.

[21] S. Riedel, L. Yao, A. Mccallum, and B. M. Marlin, "Relation extraction with matrix factorization and universal schemas," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Language Technol.*, 2013, pp. 74–84.

[22] M. Nickel, V. Tresp, and H.-P. Kriegel, "Factorizing YAGO: Scalable machine learning for linked data," in *Proc. 21st Int. Conf. World Wide Web*, 2012, pp. 271–280.

[23] Q. Wang, B. Wang, and L. Guo, "Knowledge base completion using embeddings and rules," in *Proc. 24th Int. Joint Conf. Artif. Intell.*, 2015, pp. 1859–1865.

[24] Z. Wei, J. Zhao, K. Liu, Z. Qi, Z. Sun, and G. Tian, "Large-scale knowledge base completion: Inferring via grounding network sampling over selected instances," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, 2015, pp. 1331–1340.

[25] S. Guo, Q. Wang, L. Wang, B. Wang, and L. Guo, "Semantically smooth knowledge graph embedding," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Language Process.*, 2015, pp. 84–94.

[26] R. Xie, Z. Liu, and M. Sun, "Representation learning of knowledge graphs with hierarchical types," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 2965–2971.

[27] Y. Lin, Z. Liu, H. Luan, M. Sun, S. Rao, and S. Liu, "Modeling relation paths for representation learning of knowledge bases," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2015, pp. 705–714.

[28] K. Guu, J. Miller, and P. Liang, "Traversing knowledge graphs in vector space," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2015, pp. 318–327.

[29] K. Toutanova, V. Lin, W.-t. Yih, H. Poon, and C. Quirk, "Compositional learning of embeddings for relation paths in knowledge base and text," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 1434–1444.

[30] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph and text jointly embedding," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2014, pp. 1591–1601.

[31] H. Zhong, J. Zhang, Z. Wang, H. Wan, and Z. Chen, "Aligning knowledge and text embeddings by entity descriptions," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2015, pp. 267–272.

[32] R. Xie, Z. Liu, J. Jia, H. Luan, and M. Sun, "Representation learning of knowledge graphs with entity descriptions," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 2659–2665.

[33] Z. Wang and J. Li, "Text-enhanced representation learning for knowledge graph," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 1293–1299.

[34] S. Guo, Q. Wang, L. Wang, B. Wang, and L. Guo, "Jointly embedding knowledge graphs and logical rules," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2016, pp. 192–202.

[35] T. Rocktäschel, S. Singh, and S. Riedel, "Injecting logical background knowledge into embeddings for relation extraction," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Language Technol.*, 2015, pp. 1119–1129.

[36] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A review of relational machine learning for knowledge graphs," *Proc. IEEE*, vol. 104, no. 1, pp. 11–33, 2016.

[37] N. Lao and W. W. Cohen, "Relational retrieval using a combination of path-constrained random walks," *Mach. Learn.*, vol. 81, no. 1, pp. 53–67, 2010.

[38] M. Gardner and T. Mitchell, "Efficient and expressive knowledge base completion using subgraph feature extraction," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2015, pp. 1488–1498.

[39] Q. Wang, J. Liu, Y. Luo, B. Wang, and C.-Y. Lin, "Knowledge base completion via coupled path ranking," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 1308–1318.

[40] M. Richardson and P. Domingos, "Markov logic networks," *Mach. Learn.*, vol. 62, no. 1-2, pp. 107–136, 2006.

[41] A. Kimmig, S. Bach, M. Broecheler, B. Huang, and L. Getoor, "A short introduction to probabilistic soft logic," in *Proc. NIPS Workshop Probab. Program.: Foundations Appl.*, 2012, pp. 1–4.

[42] J. Pujara, H. Miao, L. Getoor, and W. W. Cohen, "Using semantics and statistics to turn data into knowledge," *AI Mag.*, vol. 36, no. 1, pp. 65–74, 2015.

[43] R. A. Horn, "The Hadamard product," in *Proc. Symp. Appl. Math.*, vol. 40, 1990, pp. 87–169.

[44] T. A. Plate, "Holographic reduced representations," *IEEE Trans. Neural Netw.*, vol. 6, no. 3, pp. 623–641, 1995.

[45] S. He, K. Liu, G. Ji, and J. Zhao, "Learning to represent knowledge graphs with Gaussian embedding," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, 2015, pp. 623–632.

[46] H. Xiao, M. Huang, and X. Zhu, "TransG: A generative model for knowledge graph embedding," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 2316–2325.

[47] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Language Technol.*, 2013, pp. 746–751.

[48] H.-G. Yoon, H.-J. Song, S.-B. Park, and S.-Y. Park, "A translation-based knowledge graph embedding preserving logical property of relations," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Language Technol.*, 2016, pp. 907–916.

[49] D. Q. Nguyen, K. Sirts, L. Qu, and M. Johnson, "STransE: A novel embedding model of entities and relationships in knowledge bases," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Language Technol.*, 2016, pp. 460–466.

[50] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Language Process.*, 2015, pp. 687–696.

[51] G. Ji, K. Liu, S. He, and J. Zhao, "Knowledge graph completion with adaptive sparse transfer matrix," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 985–991.

[52] M. Fan, Q. Zhou, E. Chang, and T. F. Zheng, "Transition-based knowledge graph embedding with relational mapping properties," in *Proc. 28th Pacific Asia Conf. Language, Inf., Comput.*, 2014, pp. 328–337.

[53] H. Xiao, M. Huang, and X. Zhu, "From one point to a manifold: Knowledge graph embedding for precise link prediction," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 1315–1321.

[54] J. Feng, M. Zhou, Y. Hao, M. Huang, and X. Zhu, "Knowledge graph embedding by flexible translation," in *arXiv:1505.05253*, 2015.

[55] H. Xiao, M. Huang, Y. Hao, and X. Zhu, "TransA: An adaptive approach for knowledge graph embedding," in *arXiv:1509.05490*, 2015.

[56] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "Joint learning of words and meaning representations for open-text semantic parsing," in *Proc. 15th Int. Conf. Artif. Intell. Statist.*, 2012, pp. 127–135.

[57] A. Bordes, J. Weston, R. Collobert, and Y. Bengio, "Learning structured embeddings of knowledge bases," in *Proc. 25th AAAI Conf. Artif. Intell.*, 2011, pp. 301–306.

[58] S. Kullback, *Information theory and statistics*. Courier Corporation, 1997.

[59] T. Jebara, R. Kondor, and A. Howard, "Probability product kernels," *J. Mach. Learn. Res.*, vol. 5, no. Jul, pp. 819–844, 2004.

[60] D. J. Aldous, "Exchangeability and related topics," in *École d'Été de Probabilités de Saint-Flour, XIII-1983*, 1985, pp. 1–198.

[61] D. M. Blei, T. L. Griffiths, and M. I. Jordan, "The nested Chinese restaurant process and Bayesian nonparametric inference of topic hierarchies," *J. ACM*, vol. 57, no. 2, pp. 7:1–7:30, 2010.

[62] M. Nickel, L. Rosasco, and T. Poggio, "Holographic embeddings of knowledge graphs," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 1955–1961.

[63] Q. Liu, H. Jiang, A. Evdokimov, Z.-H. Ling, X. Zhu, S. Wei, and Y. Hu, "Probabilistic reasoning via deep learning: Neural association models," in *arXiv:1603.07704*, 2016.

[64] A. García-Durán, A. Bordes, and N. Usunier, "Effective blending of two and three-way interactions for modeling multi-relational

data," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2014, pp. 434–449.

[65] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," in *Proc. Int. Conf. Learn. Represent.*, 2015.

[66] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 2071–2080.

[67] K. Hayashi and M. Shimbo, "On the equivalence of holographic and complex embeddings for link prediction," in *arXiv:1702.05563*, 2017.

[68] H. Liu, Y. Wu, and Y. Yang, "Analogical inference for multi-relational embeddings," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 2168–2178.

[69] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang, "Knowledge vault: A web-scale approach to probabilistic knowledge fusion," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 601–610.

[70] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, 2009.

[71] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," in *arXiv:1207.0580*, 2012.

[72] L. Drumond, S. Rendle, and L. Schmidt-Thieme, "Predicting RDF triples in incomplete knowledge bases with tensor factorization," in *Proc. 27th Annu. ACM Symp. Appl. Comput.*, 2012, pp. 326–331.

[73] G. Bouchard, S. Singh, and T. Trouillon, "On approximate reasoning capabilities of low-rank vector spaces," in *AAAI Spring Symp. Knowl. Represent. Reasoning*, 2015.

[74] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Statist.*, pp. 400–407, 1951.

[75] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[76] T. Long, R. Lowe, J. C. K. Cheung, and D. Precup, "Leveraging lexical resources for learning entity embeddings in multi-relational data," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 112–117.

[77] D. Krompaß, S. Baier, and V. Tresp, "Type-constrained representation learning in knowledge graphs," in *Proc. 14th Int. Semantic Web Conf.*, 2015, pp. 640–655.

[78] M. Nickel and V. Tresp, "Logistic tensor factorization for multi-relational data," in *Proc. ICML Workshop Structured Learn.: Inferring Graphs Structured Unstructured Inputs*, 2013.

[79] P. Miettinen, "Boolean tensor factorizations," in *Proc. 11th IEEE Int. Conf. Data Mining*, 2011, pp. 447–456.

[80] D. Erdos and P. Miettinen, "Discovering facts with Boolean tensor Tucker decomposition," in *Proc. 22nd ACM Int. Conf. Inf. Knowl. Manage.*, 2013, pp. 1569–1572.

[81] J. D. Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[82] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.

[83] T. Franz, A. Schultz, S. Sizov, and S. Staab, "Triplerank: Ranking semantic web data by tensor decomposition," in *Proc. 8th Int. Semantic Web Conf.*, 2009, pp. 213–228.

[84] T. Van de Cruys, T. Poibeau, and A. Korhonen, "A tensor-based factorization model of semantic compositionality," in *Proc. Conf. North Amer. Chapter Assoc. Computat. Linguistics: Human Language Technol.*, 2013, pp. 1142–1151.

[85] K.-W. Chang, W.-t. Yih, and C. Meek, "Multi-relational latent semantic analysis," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2013, pp. 1602–1612.

[86] R. West, E. Gabrilovich, K. Murphy, S. Sun, R. Gupta, and D. Lin, "Knowledge base completion via search-based question answering," in *Proc. 23rd Int. Conf. World Wide Web*, 2014, pp. 515–526.

[87] S. Guo, Q. Wang, B. Wang, L. Wang, and L. Guo, "SSE: Semantically smooth embedding for knowledge graphs," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 4, pp. 884–897, 2017.

[88] M. Fan, D. Zhao, Q. Zhou, Z. Liu, T. F. Zheng, and E. Y. Chang, "Distant supervision for relation extraction with matrix completion," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics*, 2014, pp. 839–849.

[89] V. Tresp, Y. Huang, M. Bundschus, and A. Rettinger, "Materializing and querying learned knowledge," in *Proc. 1st ESWC Workshop Inductive Reasoning Mach. Learn. Semantic Web*, 2009.

[90] Y. Huang, V. Tresp, M. Nickel, A. Rettinger, and H.-P. Kriegel, "A scalable approach for statistical learning in semantic graphs," *Semantic Web*, vol. 5, no. 1, pp. 5–22, 2014.

[91] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Adv. Neural Inf. Process. Syst.*, 2001, pp. 585–591.

[92] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[93] K.-W. Chang, W.-t. Yih, B. Yang, and C. Meek, "Typed tensor decomposition of knowledge bases for relation extraction," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2014, pp. 1568–1579.

[94] Z. Hu, P. Huang, Y. Deng, Y. Gao, and E. Xing, "Entity hierarchy embedding," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Language Process.*, 2015, pp. 1292–1300.

[95] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[96] T. Zhou, J. Ren, M. Medo, and Y.-C. Zhang, "Bipartite network projection and personal recommendation," *Phys. Rev. E*, vol. 76, no. 4, 2007.

[97] A. García-Durán, A. Bordes, and N. Usunier, "Composing relationships with translations," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2015, pp. 286–290.

[98] A. Neelakantan, B. Roth, and A. McCallum, "Compositional vector space models for knowledge base completion," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Language Process.*, 2015, pp. 156–166.

[99] R. Das, A. Neelakantan, D. Belanger, and A. McCallum, "Chains of reasoning over entities, relations, and text using recurrent neural networks," in *arXiv:1607.01426*, 2016.

[100] Y. Luo, Q. Wang, B. Wang, and L. Guo, "Context-dependent knowledge graph embedding," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2015, pp. 1656–1661.

[101] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[102] D. Zhang, B. Yuan, D. Wang, and R. Liu, "Joint semantic relevance learning with text data and graph knowledge," in *Proc. 3rd Workshop Continuous Vector Space Models Compositionality*, 2015, pp. 32–40.

[103] H. Xiao, M. Huang, and X. Zhu, "SSP: Semantic space projection for knowledge graph embedding with text descriptions," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017.

[104] M. Fan, Q. Zhou, T. F. Zheng, and R. Grishman, "Distributed representation learning for knowledge graphs with entity descriptions," *Pattern Recognition Letters*, to appear, 2017.

[105] J. Xu, K. Chen, X. Qiu, and X. Huang, "Knowledge graph representation with jointly structural and textual encoding," in *arXiv:1611.08661*, 2016.

[106] L. Dehaspe and H. Toivonen, "Discovery of frequent DATALOG patterns," *Data Mining knowl. Discovery*, vol. 3, no. 1, pp. 7–36, 1999.

[107] S. Muggleton, "Inverse entailment and Progol," *New Generation Comput.*, vol. 13, no. 3, pp. 245–286, 1995.

[108] L. A. Galárraga, C. Teflioudi, K. Hose, and F. M. Suchanek, "AMIE: Association rule mining under incomplete evidence in ontological knowledge bases," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 413–422.

[109] ——, "Fast rule mining in ontological knowledge bases with AMIE+," *VLDB J.*, vol. 24, no. 6, pp. 707–730, 2015.

[110] P. Hájek, *The metamathematics of fuzzy logic*. Kluwer, 1998.

[111] T. Demeester, T. Rocktäschel, and S. Riedel, "Lifted rule injection for relation embeddings," in *Proc. Conference Empirical Methods Natural Language Process.*, 2016, pp. 1389–1399.

[112] T. Rocktäschel, M. Bošnjak, S. Singh, and S. Riedel, "Low-dimensional embeddings of logic," in *Proc. ACL Workshop Semantic Parsing*, 2014, pp. 45–49.

[113] W. Y. Wang and W. W. Cohen, "Learning first-order logic embeddings via matrix factorization," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 2132–2138.

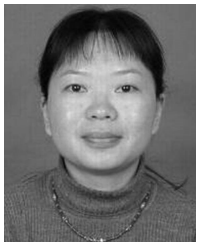
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2017.2754499, IEEE Transactions on Knowledge and Data Engineering

[114] Y. Lin, Z. Liu, and M. Sun, "Knowledge representation learning with entities, attributes and relations," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 2866–2872.
[115] T. Jiang, T. Liu, T. Ge, L. Sha, S. Li, B. Chang, and Z. Sui, "Encoding temporal information for time-aware link prediction," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2016, pp. 2350–2354.
[116] C. Esteban, V. Tresp, Y. Yang, S. Baier, and D. Krompaß, "Predicting the co-evolution of event and knowledge graphs," in *Proc. 19th Int. Conf. Inf. Fusion*, 2016, pp. 98–105.
[117] R. Trivedi, H. Dai, Y. Wang, and L. Song, "Know-Evolve: Deep temporal reasoning for dynamic knowledge graphs," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 3462–3471.
[118] D. R. Cox and P. A. W. Lewis, "Multivariate point processes," in *Proc. 6th Berkeley Symp.*, vol. 3, 1972, pp. 401–448.
[119] J. Feng, M. Huang, Y. Yang, and x. zhu, "GAKE: Graph aware knowledge embedding," in *Proc. 26th Int. Conf. Comput. Linguistics*, 2016, pp. 641–651.
[120] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 701–710.
[121] X. Jiang, V. Tresp, Y. Huang, and M. Nickel, "Link prediction in multi-relational graphs using additive models," in *Proc. Int. Conf. Semantic Technol. Meet Recommender Syst. Big Data*, 2012, pp. 1–12.
[122] M. Nickel, X. Jiang, and V. Tresp, "Reducing the rank in relational factorization models by including observable patterns," in *Adv. Neural Inf. Process. Syst.*, 2014, pp. 1179–1187.
[123] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic Web*, vol. 8, no. 3, pp. 489–508, 2017.
[124] P. Singla and P. Domingos, "Entity resolution with Markov logic," in *Proc. 6th Int. Conf. Data Mining*, 2006, pp. 572–582.
[125] M. Mintz, S. Bills, R. Snow, and D. Jurafsky, "Distant supervision for relation extraction without labeled data," in *Proc. Joint Conf. 47th Annu. Meeting ACL 4th Int. Joint Conf. Natural Language Process. AFNLP*, 2009, pp. 1003–1011.
[126] S. Riedel, L. Yao, and A. McCallum, "Modeling relations and their mentions without labeled text," *Mach. Learn. Knowl. Discovery Databases*, pp. 148–163, 2010.
[127] X. Jiang, Q. Wang, P. Li, and B. Wang, "Relation extraction with multi-instance multi-label convolutional neural networks," in *Proc. 26th Int. Conf. Comput. Linguistics*, 2016, pp. 1471–1480.
[128] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations Comput. Math.*, vol. 9, no. 6, pp. 717–772, 2009.
[129] J. Berant, A. Chou, R. Frostig, and P. Liang, "Semantic parsing on Freebase from question-answer pairs," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2013, pp. 1533–1544.
[130] A. Fader, L. Zettlemoyer, and O. Etzioni, "Paraphrase-driven learning for open question answering," in *Proc. 51st Annu. Meeting Assoc. Comput. Linguistics*, 2013, pp. 1608–1618.
[131] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, 2009.
[132] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: A heterogeneous information network approach," in *Proc. 7th ACM Int. Conf. Web Search Data Mining*, 2014, pp. 283–292.
[133] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 353–362.

Quan Wang received her PhD degree in probability and statistics from Peking University, in 2013. She is currently an associate professor at the Institute of Information Engineering, Chinese Academy of Sciences. Her research interests include information retrieval, machine learning, and natural language processing.

Zhendong Mao received his PhD degree in computer application technology from the Institute of Computing Technology, Chinese Academy of Sciences, in 2014. He is currently an assistant professor at the Institute of Information Engineering, Chinese Academy of Sciences. His research interests are in the fields of computer vision, machine learning, and artificial intelligence, in particular, on feature extraction and knowledge representation.

Bin Wang received his PhD degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences. He is currently a professor at the Institute of Information Engineering, Chinese Academy of Sciences. His research interests include information retrieval and natural language processing.



Li Guo is a professor at the Institute of Information Engineering, Chinese Academy of Sciences. She is also the chairman in the Intelligent Information Processing Research Center, Institute of Information Engineering, Chinese Academy of Sciences. Her research interests include data stream management systems and information security.