IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 10, 2018, accepted October 4, 2018, date of publication October 12, 2018, date of current version October 31, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2875066

Triple Context-Based Knowledge Graph Embedding

HUAN GAO, JUN SHI, GUILIN QI, AND MENG WANG
Computer Science and Engineering Institute, Southeast University, Nanjing 211100, China
Corresponding author: Huan Gao (gh@seu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61762063.

ABSTRACT Knowledge graph embedding aims to represent entities and relations of a knowledge graph in continuous vector spaces. It has increasingly drawn attention for its ability to encode semantics in low dimensional vectors as well as its outstanding performance on many applications, such as question answering systems and information retrieval tasks. Existing methods often handle each triple independently, without considering context information of a triple in the knowledge graph, such an information can be useful for inference of new knowledge. Moreover, the relations and paths between an entity pair also provide information for inference. In this paper, we define a novel context-dependent knowledge graph representation model named triple-context-based knowledge embedding, which is based on the notion of *triple context* used for embedding entities and relations. For each triple, the triple context is composed of two kinds of graph structured information: one is a set of neighboring entities along with their outgoing relations, the other is a set of relation paths which contain a pair of target entities. Our embedding method is designed to utilize the triple context of each triple while learning embeddings of entities and relations. The method is evaluated on multiple tasks in the paper. Experimental results reveal that our method achieves significant improvements over the state-of-the-art methods.

INDEX TERMS Knowledge graph embedding, semantic Web, link prediction.

I. INTRODUCTION

Knowledge graphs (KGs) contain rich resources that represent human knowledge in the world. Specifically, KGs describe the domain knowledge about entities, relations and their attributes. In a knowledge graph, each fact is a triple of the form (h, r, t), indicating that head entity h and tail entity t are connected with a relationship named r, e.g., *(Pablo Picasso, nationality, Spain)*. In recent years, large-scale KGs such as NELL [29], YAGO [28] and DBpedia [27] are becoming important resources. Plenty of AI-related tasks, such as question answering and information extraction systems, benefit from the domain knowledge of KGs.

While typical KGs are often carefully created, the large-scale KGs often contain billions of facts (triples). However, they are woefully incomplete and an incomplete KG cannot effectively support the KG-based applications. Hence, there is an increasing interest in knowledge completion by means of *knowledge graph embedding*. It aims to effectively encode a relational knowledge base into a low dimensional continuous vector space. Each entity or relation is represented as a vector or a matrix which contains rich semantic information and can enhance knowledge acquisition and inference significantly [15]. Early work [3] shows that embeddings are more tractable and scalable to operate than the original symbolic KG. Recent studies reveal that knowledge graph representation learning methods have good power to learn the vector embeddings of entitles and relations, which can predict whether the unknown relational facts are true or not. The typical KG embedding models, such as TransE [7], TransH [8] and TransR [16], model a relation as a translation between an entity pair. Some other models, like RESCAL [5], DistMult [19], HolE [10] and ComplEx [22], adopt different compositional operations to capture the rich interactions of embeddings.

Despite the success of previous methods in KG embedding, most of the existing models only learn embeddings from triples. For example, in a typical translation-based method TransE [7], the basic idea is that when a triple is true, the embedding of its head entity h plus the embedding of its relation entity r equals to the embedding of its tail entity t. Translation-based methods achieve huge success in many tasks such as link prediction and triple classification.

Nevertheless, since structured information is ignored, TransE has some deficiencies when modeling the complex relations such as reflexive, transitive, 1-to-N, N-to-1 or N-to-N relations. In reality, a lot of information is implicit in the surrounding structure of a triple which contains neighbor entities and relation paths and could be viewed as a directed subgraph of a KG. Most previous models have always ignored this kind of structured information which can be further used for inference over the KG. Moreover, in a KG, there are many neighbor entities around each entity which may interact with different relations. The neighbor entities are usually utilized to provide the implied inter-relation between the entity and their neighbors. What's more, the head entity and tail entity are connected with a set of multiple-step relation paths which indicate the semantic relationships and reflect the complicated inference pattern. For example, to decide whether *Steven Curry* plays for *Golden States Warriors*, Fig.1 displays a relation path like $a \xrightarrow{teammate} b \xrightarrow{play\_for} c$ which indicates that there may be a relation *play_for* between a and c.

There have been a few works that incorporate graph structures into KG embedding models. Models in [12] and [14] introduce the relation path information into model learning. The study shows that the relation paths between entities provide useful information and improve the performance of KG completion. These methods only consider relation information while ignoring other structured information which contains rich clues for inference. GAKE [11] leverages structured information and utilizes three types of context for embedding the KG. However, GAKE views both entities and relations as the target nodes, in which way the semantic relations between entities and relation are ignored and their embedding cannot be learned well.

Aiming to utilize the structured information, we propose a novel knowledge graph embedding model named Triple-Context-based Knowledge Embedding (TCE). TCE aims to improve modeling the complex relations between two entities and leverage a novel context which takes the surrounding structured information of a triple into consideration. Our paper comes up with a novel graph-based context named triple context which is composed of neighbor context and path context. A neighbor context consists of directed outgoing relations of an entity and its neighbor entities which are linked by the outgoing relations. References [4] and [12] prove that the multiple-step relation paths reflect the semantic relation between an entity pair and is helpful to deal with complex relationships. In TCE, the same concept is chosen to build the path context. The two kinds of context are integrated to build a triple context that can improve the effectiveness of TCE for predicting missing entities and relations given by their triple contexts. With this goal in mind, a novel score function is introduced to evaluate the likelihood of a triple and to learn embeddings. In order to describe TCE more detail, we extend our CIKM short paper [30]. Compared with our short paper, we introduce a new method to calculate the score function of TCE and add a new experiment for relation prediction, entity prediction for Hits@N and using different score functions for link prediction.

The main contributions of our work are summarized as follows:

1) Instead of a set of independent triples, the local structure around a triple in the KG is utilized to study the representation of knowledge;

2) Based on the triple context, the study proposes a novel knowledge graph embedding model, TCE, which can easily utilize the triple context well;

3) The experiments show that our model outperforms other baselines impressively in some aspects.

The rest of the paper is organized as follows. In Section II, some related work is introduced. Section III expounds on the knowledge graph and the triple context in detail. The model and the objective function of TCE are described in Section IV. Section V provides the details of our experimental settings such as datasets and evaluation protocol. The experimental results are analyzed thoroughly by comparing the method for the study with previous ones. Finally, the concluding remarks are provided in Section VI.

II. RELATED WORK

In this section, we briefly summarize the most relevant works. Generally, there are two topics which are closely related to our study, one is knowledge graph embedding and the other is graph embedding.

A. KNOWLEDGE GRAPH EMBEDDING

In recent years, *knowledge graph embedding* has been widely studied, see [5]–[8], [16]. It aims to effectively encode a relational knowledge base into a low dimensional continuous vector space and achieves success on relational learning tasks like link prediction [13] and triple classification [6]. Various techniques have been devised for this task, for instance, translation-based models which take relations as translating operations between head and tail entities. One of the most typical translation-based models is TransE which assumes a head entity can be translated to a tail entity by a relationship. TransE can well balance the effectiveness and efficiency compared to traditional methods, while the over-simplified translation assumption constrains the performance when dealing with complicated relations. To deal with this deficiency, several improved models are proposed. In order to find the most related relation between two entities, TransA [13] sets different weights according to different axis directions. In TransH [8], each entity is projected into the relation-specific hyperplane that is perpendicular to the relationship embedding. TransD [18] and TransR [16] still follow the principle of TransH, The two models project entities into relation-specific spaces in order to process the complex relations. Moreover, the translation assumption only focuses on the local information in triples such as a single triple, which may fail to make full use of global graph information in KGs.

Unlike the aforementioned translation-based models, ComplEx [22] and DistMult [17] are extensions of

RESCAL [5]. DistMult [17] constructs a proof bilinear model to describe triples and restrict the matrices representing relations to diagonal matrices. However, it also has the problem that the scores of *(h,r,t)* and *(t,r,h)* are the same. To solve this problem, ComplEx [22] selects complex numbers instead of real numbers and takes the conjugate of the embedding of the tail entity before calculating the bilinear map. ProjE [20] and NTN [6] present the neural networking models which contain more parameters and also attract much attention, but they tend to overfit the training data. In addition, some work shows that integrating additional information will improve the performance, such as text [9] and entity type [23].

In the above knowledge graph models, triples in KGs are independent and the interrelations of each triple are ignored, and a KG can be viewed as a graph with its graph-structured information which gives the power in knowledge inference. PTransE [12] is an extending model of TransE to model a path-based representation learning model. PTransE utilizes connected relational facts between entity pairs instead of only considering the relation between two entities. However, it is rather hard for PTransE to model the complex context for diverse entities and relations because it merely uses relation paths as new relations in the KG, ignoring other information contained in graph structures. The most relevant model for the research is GAKE [11]. In GAKE, it defines three types of graph context which contain different KGs structured information for representation learning. The score function of GAKE only takes into account the connection between the target entity or relation and context. Thus, the internal relation of a triple is ignored in which way the structured information is still lost. All these studies demonstrate that structured information is able to enhance the knowledge graph embedding.

B. GRAPH EMBEDDING

Analyzing graph structured information of large networks has been attracting increasing attention from both academia and industry. For example, DeepWalk [24] combined random walk and skip-gram to learn network representations. LINE [25] is a network embedding model that extends DeepWalk with a more sophisticated random walk or breadth-first search schemes to preserve both the local and global network structures. CANE [26] applied the text information and the mutual attention mechanism to learn context-aware embedding.

Graph embedding models and knowledge graph embedding models are similar in the sense that both utilize the network structures to learn the continuous representations for vertices. However, there exists a vast difference between the two kinds of embedding models. The graph embedding model cannot directly learn the embeddings of entities and relations in KGs. It is because the graph embedding models aim to learn the embeddings for vertices, but the knowledge graph embedding models need to learn the embeddings of both vertices and edges which are entities and relations in knowledge graph respectively. Moreover, in graph embedding models, it is assumed that the connected vertices should be similar. However, it is not held in a knowledge graph. The similarity of connected vertices depends on the semantic and the local structure. For instance, in triple (*Barack_Obama*, *Birth_Place*, *Hawaii*), there are no common attributes between the head entity *Barack_Obama* and tail entity *Hawaii*.

Our paper aims to propose a KG embedding model and propose a graph-based model TCE which can learn the latent representation of entities and relations by formulating a triple context as a graph context. TCE leverages the structured information of the triple context to obtain better embeddings.

III. PRELIMINARIES

In the following part, some necessary notations are firstly introduced. A knowledge graph $\mathcal{K}$ is made up by a set of triples. In $\mathcal{K}$, each triple is denoted as (h, r, t) where r is the relation between head entity h and the tail entity t. All head entities and tail entities in $\mathcal{K}$ belong to a set of entities $\mathcal{E}$. Similarly, all relations in $\mathcal{K}$ belong to a set of relations $\mathcal{R}$. Our model aims to learn and embed all entities in $\mathcal{E}$ and relations which are denoted as $\mathcal{R}$ in a d-dimensional space $\mathbb{R}^d$. The following part gives a brief introduction of knowledge graph and then defines the triple context.

A. KNOWLEDGE GRAPH

The traditional view is a KG is constituted of a set of triples, which describe a set of independent facts. Intending to utilize the structured information of a KG, we shall consider a KG as a directed graph. All entities and relations in the KG are treated as vertices and directed edges respectively. More formally, let $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ be a directed graph of a KG $\mathcal{K}$. In $\mathcal{G}$, the vertices $\mathcal{V}$ denote the entities in $\mathcal{E}$ and the set of edges $\mathcal{A}$ represents a directed relation in $\mathcal{R}$ from the head entity to the tail entity.

In order to utilize the structured information, we build a directed graph from a set of triples(or facts). The procedure of building the directed graph is as follows: given a set of triples of $\mathcal{K}$, for each triple (h, r, t) in $\mathcal{K}$, where h and t indicate head entity and tail entity respectively. In the beginning, two vertices v_i and v_j are created, the two vertices are corresponding to h and t in the triple. After that, the relation r which connects h and t is adopted to generate a directed edge e. It is worth noting that the reverse relation r^{-1} from v_j to v_i is a common trick which is usually applied to *back translation* in machine translation. The greatest advantage of the reverse relation is that it can improve the performance by fully utilizing the structured information of a KG. After all entities and relations in $\mathcal{K}$ are contained in the directed graph $\mathcal{G}$, the above process of building a graph comes to an end. Figure 1 shows the example of knowledge which is composed of 9 entities, 10 relations, and 10 facts.

B. TRIPLE CONTEXT

1) NEIGHBOR CONTEXT

A neighbor context of an entity is the surroundings of it in KGs. It is the local structure that interacts mostly with the entity. Specifically, given an entity e, the neighbor context

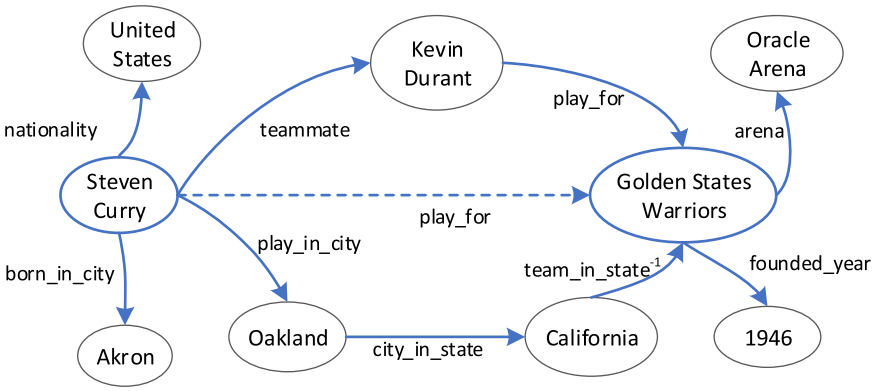


FIGURE 1. An example of the *triple context* of a triple (*Steven Curry*, *play_for*, *Golde States Warriors*) in a knowledge graph.

of e is a set $C_n(e) = \{(r,t)|\forall r,t,(e,r,t) \in \mathcal{K}\}$, where r is an outgoing edge (relation) from e and t is the entity it reaches through r. In other words, the neighbor context of e is all the *relation-tail* pairs appearing in triples with e as the head. For example, as shown in Figure. 1, the neighbor context of entity *Steven Curry* is $C_n(Steven\,Curry) = \{(nationality, United\ States), (born\_in\_city, Akron)\}$. The appearance of an entity is predicted based on its neighbor context in our model, as a measurement of the compatibility of the entity and its neighbor context.

2) PATH CONTEXT

A path context is a piece of important information for modeling the relation and capturing complicated inference patterns among relations in KGs. For each triple, its path context consists of multiple-step relation paths that start from the head entities to the tail entity. More formally, given a pair of entities (h,t), the path context of (h,t) is a set $C_p(h,t) = \{p_i|\forall r_{m_1},\cdots,r_{m_i},e_1,\cdots,e_{m_i-1},\ p_i = (r_{m_1},\cdots,r_{m_i}),(h,r_{m_1},e_1) \in \mathcal{K},\cdots,(e_{m_i-1},r_{m_i},t) \in \mathcal{K}\}$, where p_i is a list of relations (labeled edges) through which it can traverse from h to t, m_i is the length of path p_i. In Figure. 1, the path context between *Steven Curry* and *Golden States Warriors* is $C_p(h,t) = \{(teammate, play\_for), (play\_in\_city, city\_in\_state, tea\ m\_in\_state^{-1})\}$.

IV. TCE

In this section, TCE is described for learning the representation of a given KG. TCE inputs a set of triples with their triple context and project entities and relations into low dimensional vectors.

A. MOTIVATION OF TCE

Despite the success of translation-based models and structure-based models in KG embedding, they are still not good at dealing with complex relations. The complex relation between two kinds of models is analyzed to illustrate the causes.

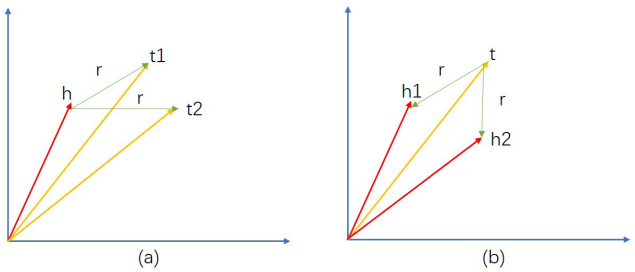


FIGURE 2. Illustrations of TransE for complex relations.

1) For each entity in KG, translation-based models are hard to find the correct tail entity for such complex relations. Fig. 2 illustrates this problem in TransE for complex relations. There exist two triples and r is a 1-to-N relation. Since all entities and relations are embedded in the same space, $h-t_1 = r \neq h-t_2$ can be derived from the principle $h+r \approx t$.

2) GAKE is a structure-based embedding model and the most relevant model for our work. It leverages three types of graph context for processing link prediction. GAKE considers the connection between the subject and context to obtain the embedding of the subject that leads to predict complex relations in failure. Similarly, there exist two triples and r is a 1-to-N relation in Fig. 3(a). The context of head entities h is shown in Fig. 3(b). In Fig. 3(b), the context of h is the same for the two triples (h,r,t_1) and (h,r,t_2), thus the model is hard to predict tail entity t by the given head entity h and the relation r. The main reason is the embedding of h for the two triples is the same. By the score function in GAKE of $p(t|h,r)$, we will obtain $t_1 \approx t_2$. Fig. 3(c) and Fig. 3(d) show the similar phenomenon of the N-to-1 relation. By given t and r, GAKE will have $h_1 \approx h_2$.

Targeting at solving the above problem, TCE builds the context for the triple that can help TCE to distinguish complex relations. Meanwhile, the internal connection of a triple is also considered in TCE to improve the performance of TCE. Fig. 4(a) and Fig. 4(c) are the same cases in Fig. 3(a) and Fig. 3(c). In Fig. 4(b), the triple contexts of the two triple

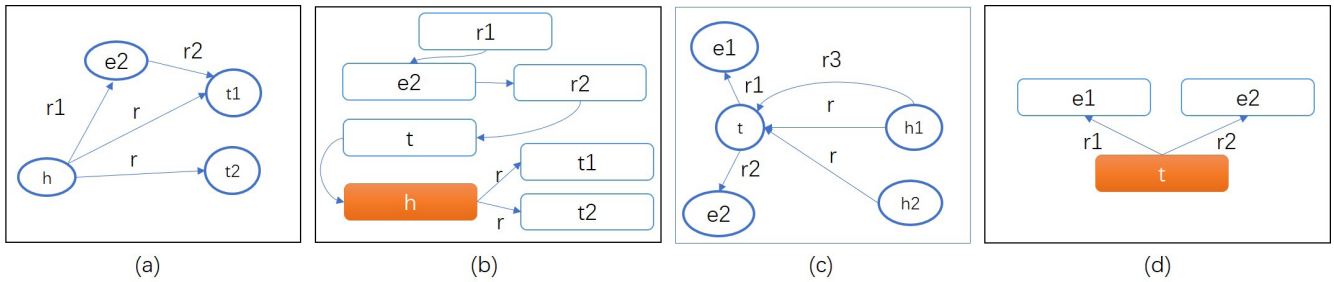


FIGURE 3. Illustrations of GAKE for complex relations.

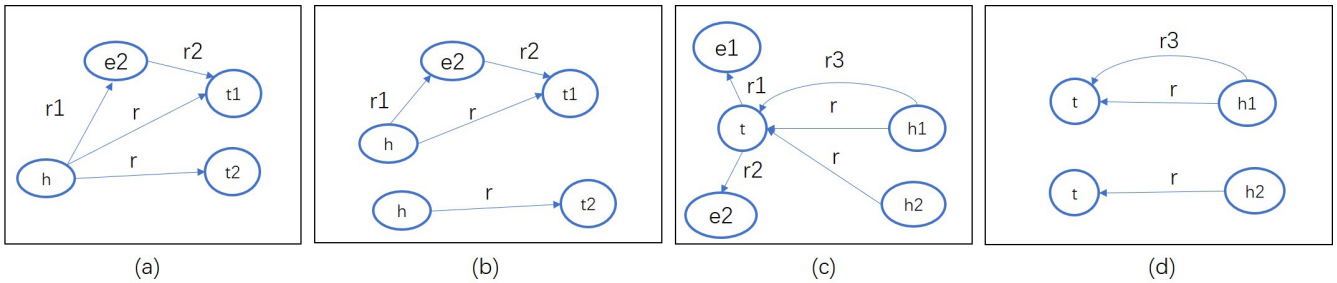


FIGURE 4. Illustrations of TCE for complex relations.

are different and can be viewed as an additional information to help TCE to handle the complex relations. For $(h, r, t1)$, the triple context $C(h, r, t1)$ is composed of a path context $C_p(h, t1) = \{(r1, r2)\}$ and the triple $(h, r, t1)$ itself. For triple $(h, r, t2)$, its triple context $C(h, r, t2)$ only contains the triple $(h, r, t2)$ itself. Thus, the the probabilities of $P((h, r, t1)|C(h, r, t1))$ and $P((h, r, t2)|C(h, r, t1))$ are different by given the triple context that can help TCE to predict the correct missing tail entities. It is similar in Fig. 4(d), the two triples $(h1, r, t)$ and $(h2, r, t)$ have different triple contexts thus the probability $P((h1, r, t)|C(h1, r, t))$ and $P((h2, r, t)|C(h1, r, t))$ are different.

B. MODELING TCE

Till the end of the section III, the introduction of neighbor context and path context have been completed. For each triple in a KG, its triple context is made up of a neighbor context of head entity h, path context between the entity pair (h, t) (path context contains the triple itself), which can be formalized as:

$$C(h, r, t) = C_n(h) \cup C_p(h, t). \tag{1}$$

For each triple, its triple context can be considered to a subgraph in the KG which provides surrounding structures to make TCE aware of the information that is contained in the graph structures.

Next, the method is explained in detail. In general KG embedding models, since the model handles each triple independently, the score functions of a triple are only related to the embeddings of entities and relation in the triple. For example, TransE defines the score function as $f_{TransE}(h, r, t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_{L_1/L_2}$. In our method, a triple context is introduced in the score function. Given a candidate triple (h, r, t), we introduce a conditional probability that the triple appears given the respective triple context and all the embeddings, as the score function:

$$f(h, r, t) = P((h, r, t)|C(h, r, t); \Theta) \tag{2}$$

where $C(h, r, t)$ is the triple context of (h, r, t) and Θ is the parameter of this probability. A higher score of a triple indicates that it holds to a greater extent.

Generally, the learning objective of TCE is to predict the missing entities or relations given by their contexts. More formally, we define an objective function by maximizing the joint probability of all triples in a knowledge graph $\mathcal{K}$, which can be formulated as:

$$P(\mathcal{K}|\Theta) = \prod_{(h,r,t)\in\mathcal{K}} f(h, r, t) \tag{3}$$

For the score function in Eq. (2), the conditional probability formula is utilized to decompose the probability $P((h, r, t)|C(h, r, t); \Theta)$ as:

$$\begin{aligned} f(h, r, t) = {} & P(h|C(h, r, t); \Theta) \\ & \cdot P(t|C(h, r, t), h; \Theta) \\ & \cdot P(r|C(h, r, t), h, t; \Theta) \end{aligned} \tag{4}$$

where the evaluation of the whole triple is decomposed into three parts, which are explained in detail as follows.

The first part $P(h|C(h, r, t); \Theta)$ in Eq. (4) represents the conditional probability that h is the head entity given the triple context and all the embeddings. Since the representation of h is decided by the neighboring structures of h in the KG,

$P(h|C(h,r,t);\Theta)$ can be approximated as $P(h|C_n(h);\Theta)$, where $C_n(h)$ is the *neighbor context* of h. The approximated probability $P(h|C_n(h);\Theta)$ can be considered as the compatibility between h and its neighbor context, it is formalized as a softmax-like representation, which is also used in [1] and has been validated as follows:

$$P(h|C_n(h);\Theta) = \frac{\exp(g_n(h, C_n(h)))}{\sum_{h'\in\mathcal{E}} \exp(g_n(h', C_n(h)))} \tag{5}$$

where $g_n(\cdot,\cdot)$ is the function that describes the correlation between an arbitrary entity h' and the neighbor context of the specific entity h. $g_n(h', C_n(h))$ is defined as:

$$g_n(h', C_n(h)) = \frac{1}{|C_n(h)|} \sum_{(r,t)\in C_n(h)} \|\mathbf{h'} \circ \mathbf{r} \circ \mathbf{t}\| \tag{6}$$

where $|C_n(h)|$ is the size of neighbor context $C_n(h)$ and $\|\cdot\|$ denotes 1-norm or 2-norm in this paper. As is shown in DistMult [17], adopting multiplication as the compositional operation is superior to addition while modeling entity relations. Hence, given an entity h' and a pair (r,t) from $C_n(h)$, Hadamard product(element-wise product) is applied to compose them as the measure of their relevance. Then, we take an average of norms of all neighbor contexts is taken and the relevance between h' and $C_n(h)$ is reflected.

The second part $P(t|C(h,r,t),h;\Theta)$ in Eq. (4) is the conditional probability that t is the tail entity given the head entity h, triple context and all the embeddings. The path context between h and t measures the relatedness of them and approximate $P(t|C(h,r,t),h;\Theta)$ as $P(t|C_p(h,t),h;\Theta)$, where $C_p(h,t)$ is the *path context* between h and t. The approximated probability $P(t|C_p(h,t),h;\Theta)$ is formalized as follows:

$$P(t|C_p(h,t),h;\Theta) = \frac{\exp(g_p(t, C_p(h,t)))}{\sum_{t'\in\mathcal{E}} \exp(g_p(t', C_p(h,t)))} \tag{7}$$

where $g_p(\cdot,\cdot)$ is a function of correlation between an arbitrary entity t' and the path context of the specific entity pair (h,t), which is formalized as:

$$g_p(t', C_p(h,t)) = \frac{1}{|C_p(h,t)|} \sum_{p\in C_p(h,t)} \|\mathbf{h} \circ \mathbf{p} \circ \mathbf{t'}\| \tag{8}$$

where $\mathbf{p}$ composes all relations in p into a single vector by averaging over all their embeddings. For example, path $p_i = (r_{m_1}, \cdots, r_{m_i})$, the embedding of it is $\mathbf{p}_i = \frac{1}{m_i}(\mathbf{r}_{m_1} + \cdots + \mathbf{r}_{m_i})$. Eq. (8) has a similar meaning as Eq. (6) and it reflects the relevance between t' and $C_p(h,t)$.

The last part $P(r|C(h,r,t),h,t;\Theta)$ in Eq. (4) is the conditional probability describing that relation r holds given the triple context, head entity h, tail entity t and all the embeddings. Since h and t have been determined and triple context has been incorporated in the previous two parts, $C(h,r,t)$ can be omitted in $P(r|C(h,r,t),h,t;\Theta)$, which is then formulated as follows:

$$P(r|h,t;\Theta) = \frac{\exp(g_r(h,r,t))}{\sum_{r'\in\mathcal{R}} \exp(g_r(h,r',t))} \tag{9}$$

where $g_r(\cdot,\cdot,\cdot)$ is a function indicating the connection between relation r and entity pair (h,t). Here, we define function $g_r(h,r,t)$ as:

$$g_r(h,r,t) = \|\mathbf{h} \circ \mathbf{r} \circ \mathbf{t}\| \tag{10}$$

where $\circ$ is the Hadamard product as is introduced in Eq. (6). TCE can easily replace Hadamard product with other functions. In this paper, the energy function of TransE is taken instead of Hadamard product. Then, Eq. (6), Eq. (8) and Eq. (10) are formalized as follows:

$$g_n(h', C_N(h)) = -\frac{1}{|C_N(h)|} \sum_{(r,t)\in C_N(h)} \|\mathbf{h'} + \mathbf{r} - \mathbf{t}\| \tag{11}$$

$$g_p(t', C_P(h,t)) = -\frac{1}{|C_P(h,t)|} \sum_{p\in C_P(h,t)} \|\mathbf{h} + \mathbf{p} - \mathbf{t'}\| \tag{12}$$

$$g_r(h,r,t) = -\|\mathbf{h} + \mathbf{r} - \mathbf{t}\| \tag{13}$$

It is noteworthy that for each path p in Eq. (12) that the embedding of it is $\mathbf{p}_i = \mathbf{r}_{m_1} + \cdots + \mathbf{r}_{m_i}$. We then take the negative average of all conditions, thus the relevance between a triple, and its context is reflected.

Accordingly, the score function in Eq. (4) can be approximated as:

$$\begin{aligned} f(h,r,t) \approx\ & P(h|C_n(h);\Theta) \\ & \cdot P(t|C_p(h,t),h;\Theta) \\ & \cdot P(r|h,t;\Theta) \end{aligned} \tag{14}$$

in which way the neighbor context and the path context of a triple are incorporated. It is noteworthy that we do not use the neighbor context of entity t in our model, for the consideration that t can also be the head entity of other triples and its neighbor context may be incorporated elsewhere.

C. MODEL LEARNING

In order to facilitate computing the embedding, the score function is transformed to Eq. (14) which is the approximation through softmax form. In Eq. (14), Eq. (5), Eq. (7) and Eq. (9) represent the probability of each part in triple context and the triple itself respectively. In order to avoid high computational overhead, the negative sampling [2] is adopted in TCE to approximate full softmax function efficiently. In this way, the likelihood of the whole knowledge graph $\mathcal{K}$ is approximated via negative sampling as follows:

$$\begin{aligned} & P(\mathcal{K}|\Theta) \\ & = \prod_{(h,r,t)\in\mathcal{K}} f(h,r,t) \\ & \approx \prod_{(h,r,t)\in\mathcal{K}} P(h|C_n(h);\Theta)\cdot P(t|C_p(h,t),h;\Theta)\cdot P(r|h,t;\Theta) \\ & \approx \prod_{(h,r,t)\in\mathcal{K}} [\sigma(g_n(h,C_n(h)))\cdot \prod_{h'} \sigma(-g_n(h',C_n(h)))] \\ & \quad \cdot [\sigma(g_p(t,C_p(h,t)))\cdot \prod_{t'} \sigma(-g_p(t',C_p(h,t)))] \\ & \quad \cdot [\sigma(g_r(h,r,t))\cdot \prod_{r'} \sigma(-g_r(h,r',t))] \end{aligned} \tag{15}$$

where h', r' and t' are negative samples. Here, h' is generated by replacing the head entity with an arbitrary entity. Moreover, $\sigma(\cdot)$ is the logistic function. For the convenience of parameter optimization, Eq. (15) is transformed into the negative logarithm form.

$$\begin{aligned}\log P(\mathcal{K}|\Theta) = \sum_{(h)\in\mathcal{K}} [&\log\sigma(g_n(h, C_n(h)))\\ &+ \sum_{h'} \log\sigma(-g_n(h', C_n(h)))\\ &+ \sum_{(t)\in\mathcal{K}} \log\sigma(g_p(r, t, C_p(h, t)))\\ &+ \sum_{t'} \sigma(-g_p(r, t', C_p(h, t)))]\\ &+ \sum_{(r)\in\mathcal{K}} \log\sigma(g_r(h, r, t))\\ &+ \sum_{r'} \sigma(-g_r(h, r', t))] \end{aligned} \tag{16}$$

In a real KG, the neighbors and relation paths may be large for each triple. In order to find all neighbor entities, TCE needs to search all triples for each head entity. Similarity, if TCE need find all paths for each pair of entities, the time complexity for searching is exponentially increasing with the length of the path. A large number of neighbors and relation paths result in great computation consumption. In order to improve the computation efficiency, the number of neighbors and paths is less than two thresholds in triple context, where n_N is for neighbor context and n_P is for path context. When the neighbors or paths exceed its threshold, we sample a subset whose size equals threshold from all neighbors or paths. It is worth noting that TCE just adopts the relation paths whose length is less than 3 when the path context is sampled.

D. TRAINING TCE

Alg.1 describes the training process of TCE. For a given training triple set $\mathcal{S}$ composed of two entities $h, t \in \mathcal{V}$ and a relationship $r \in \mathcal{E}$, our model learns embeddings of the entities and the relations. The embeddings take values in $\mathbb{R}^d$ where d is a hyperparameter. All embeddings for entities and relations are firstly initialized following uniform distribution $U[-6/sqrt(d), 6/sqrt(d)]$ as suggested in TransE. At each main iteration of the algorithm, a small set of triples is sampled from the training set to serve as the training triples of minibatch. For each such triple, we then sample a single corrupted triple by randomly corrupting the head entity h or tail entity t to generate the corresponding positive and negative candidates from $\mathcal{S}$. Then for each mini-batch in the newly generated training data set, we construct the neighbor context and path context of each triple. Finally, we calculate the loss which favors high probability for the positive triples than the negative triples and renew the embedding accordingly. The algorithm is stopped based on its performance on a validation set.

Table 1 summarizes the space and time complexity of the different knowledge graph embedding mode. Here, m and n represent the number of entities and relations respectively; k signifies the dimension of entity and relation embedding space. TransE, TransH, DistMult and ComplEx only project entities and relations into a low-dimensional space. These models are efficient because of space and time complexity scale linearly with k. TransR and RESCAL model the relations as matrices and tensors respectively. Because their embedding space is scaling quadratically or cubically, The two models usually have higher complexity in both space and time. By comparison, the time complexity of TCE is better than TransR and RESCAL. However, TCE has a higher time complexity as additional information is introduced. In order to train TCE effectively, the scale of neighbors and relation paths is limited.

Algorithm 1 Training TCE

Input : A training set $\mathcal{S} = \{(h, r, t)\}$
the entity set $\mathcal{E}$ of $\mathcal{S}$
the relation set $\mathcal{R}$ of $\mathcal{S}$
the dimension of embedding k
Output: Embedding Model
begin
  initialize
  $\mathbf{r} \leftarrow uniform(-6/\sqrt{k}, 6/\sqrt{k})$ for each $r \in \mathcal{R}$
  $\mathbf{e} \leftarrow uniform(-6/\sqrt{k}, 6/\sqrt{k})$ for each entity $e \in \mathcal{E}$
  while *not reach at loop number* **do**
    foreach *batch* S_{batch} *in* $\mathcal{K}$ **do**
      foreach $(h, r, t) \in S_{batch}$ **do**
        construct a set of corrupted triples $S'_{(h,r,t)} = \{(h', r, t')\}$;
        construct the neighbor context of (h, r, t);
        construct the path context of (h, r, t);
      end
      update embedding w.r.t
      $\sum_{(h,r,t)\in S_{batch}} \nabla[\sum_{(h)\in\mathcal{K}}[\log\sigma(g_n(h, C_n(h)))+$
      $\sum_{h'} \log\sigma(-g_n(h', C_n(h))) +$
      $\sum_{(t)\in\mathcal{K}} \log\sigma(g_p(r, t, C_p(h, t))) +$
      $\sum_{t'} \sigma(-g_p(r, t', C_p(h, t)))] +$
      $\sum_{(r)\in\mathcal{K}} \log\sigma(g_r(h, r, t)) +$
      $\sum_{r'} \sigma(-g_r(h, r', t))]$
    end
  end
  return *Embedding Model*;
end

TABLE 1. Comparison of models in space and time complexity.

Model	Time Complexity	Space Complexity
TransE	$\mathcal{O}(nk + mk)$	$\mathcal{O}(k)$
TransH	$\mathcal{O}(nk + mk)$	$\mathcal{O}(k)$
TransR	$\mathcal{O}(nk + mk)$	$\mathcal{O}(lk)$
RESCAL	$\mathcal{O}(nk + mk^2)$	$\mathcal{O}(k^2)$
DistMult	$\mathcal{O}(nk + mk)$	$\mathcal{O}(k)$
ComplEx	$\mathcal{O}(nk + mk)$	$\mathcal{O}(K)$
TransH	$\mathcal{O}(nk + mk)$	$\mathcal{O}(N_n k + N_p K)$

V. EXPERIMENTS

To scrutinize the effectiveness of our model, we introduce two benchmark datasets and compares the performance of TCE with other baselines on the tasks of link prediction and relation prediction. N_n and N_p denote two thresholds for neighbor context and for path context.

A. DATASETS AND EXPERIMENT SETTINGS

1) DATASETS

We select two widely-used benchmark datasets FB15k [7] and FB15k-237 [21] for evaluation, which are extracted from a real-world KG Freebase and contain rich structured information. WN18 is not involved in our experiments because each triple in WN18 has few neighbors and relation paths. As a result, some triples cannot construct their triple contexts. FB15k has 592,213 triples with 149,51 entities and 1,345 relationships. FB15k-237 has 310,116 triples with 14,541 entities and 237 relations. FB15k-237 is derived from FB15k and filtered to remove highly redundant relations, with the goal of making the task more realistic [21]. The two datasets are further divided into training, validation, and test set respectively. Table 2 list statistics of entities in the two data sets.

TABLE 2. Statistics of the data sets.

DataSet	#Ent	#Rel	#Train	#Valid	#Test
FB15K	14,951	1,345	483,142	50,000	59,071
FB15K-237	14,541	237	272,155	17,535	20,466

2) BASELINES

We compare our model TCE with several state-of-art KG embedding models including TransE [7], TransH [8], TransR [16], CTransR [16], PTransE [12], HolE [10], ProjE [20], DistMult [17], ComplEx [22], PRA [4], ARP [32], R-GCN+ [33] and GAKE [11].

Unstructured model assumes that the head and tail entity vectors are similar. As the Unstructured model does not take the relationship into account, it cannot distinguish different relation types.

TransE [7] is a translational embedding model that models relational triples with $\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|_{L_1/L_2}$.

TransH [8] associates each relation with a relation-specific hyperplane and adopts a projection vector to project entity vectors onto that hyperplane.

TransR [16] and **CTransR** [16] extend the TransH model by means of two projection vectors and a matrix to project entity vectors into a relation-specific space respectively.

PTransE [12] is a path-based model, simultaneously considering the information and confidence level of a path in the knowledge graph.

PRA [4] learns to infer different target relations by tuning the weights associated with random walks that follow different paths through the graph.

ARP [32] designed a generic framework to combine latent and observable variable models. In our experiment, we combine RESCAL with PRA as the path features.

ProjE [20] is a neural model which applies a list-wise ranking loss and views the link prediction task as ranking .problem and can learn the joint embeddings of both entities and relations.

DistMult [17] is based on the Bilinear model where each relation is represented by a diagonal rather than a full matrix.

ComplEx [22] represents relations and entities with complex-valued embeddings and scores a triple (h, r, t) with the real part of the trilinear product of the corresponding embeddings

GAKE [11] formulates the knowledge base as a directed graph and learns embeddings for any vertices or edges by leveraging the graphs structured information.

R-GCN+ [33] is a multi-layer convolutional network model which is convenient to represent KGs as directed labeled multigraphs with entities corresponding to nodes and triples.

3) IMPLEMENTATION DETAILS

TCE is implemented in Python with the aid of TensorFlow. The source code and data will be available online. We construct KGs from the benchmark datasets through Apache TinkerPop, which is an open source graph computing framework. According to the definition of path context in Section III-B, we consider the path context following one direction. However, in some cases, the reverse version of a relation should also be considered, which may provide useful path information but not be presented in KGs. For instance, the relation path $a \xrightarrow{motherOf} b \xleftarrow{fatherOf} c$, i.e., $(a, motherOf, b)$ and $(c, fatherOf, b)$, indicates a potential relation *marriedTo* between a and c. Therefore, we add the reverse relation for each relation in KGs. For each edge labeled r from h to t in the graph, we add another edge labeled r^{-1} from t to h.

Searching all neighbors and relation paths for a triple during the learning process will affect the efficiency of the mode learning. Therefore, we generate the triple context for each triple before the model learning. During this procedure, it is found that the size of a triple context may be huge, and we can hardly make use of all context information in this case. Therefore, for both neighbor context and path context, our model sample ten neighbors randomly and ten relation paths according to the frequency of path pattern. For the triples with less than ten neighbors and relation paths in benchmark datasets, our model pad vectors filled with zeros as the neighbor context and path context.

mini-batch Adam is utilized on two datasets for training TCE model. As for parameters, the learning rate α of Adam is chosen from the range of $\{0.1, 0.01, 0.001\}$, while the dimension of embeddings k range from $\{50, 100, 200\}$, and the batch size B within the scope of $\{120, 480, 960, 1920, 4800\}$. The best configuration is determined according to the mean rank in the validation set. The optimal parameters are $\alpha = 0.001$, $k = 50$ and $B = 4800$.

B. LINK PREDICTION

Link prediction [8] is a task to complete the triple (h, r, t) when h, r or t is missing. The evaluation is made on the datasets and the link prediction task is divided into two sub-tasks: entity prediction and relation prediction.

1) EVALUATION PROTOCOL

Following the same protocol used in [7], we takes *Mean Rank* and *Hits@10* as evaluation protocols. For each test triple (h, r, t), we replace the entity h/t (relation r) with each entity e in $\mathcal{E}$ (relation r in $\mathcal{R}$) to generate *corrupted triples* and calculate the score of each triple using the score function. After ranking the scores in descending order, the rank of the correct entities (relations) is then derived. *Mean Rank* is the mean of all the predicted ranks, and *Hits@10* denotes the proportion of correct entities or relations ranked in the *top N*. Note that a corrupted triple ranking above a test triple could be valid, which should not be counted as an error. Hence, corrupted triples that already exist in KGs are filtered before ranking. The filtered version is denoted as "Filter," and the unfiltered version is represented as "Raw." The "Filter" setting is usually preferred. In both settings, a higher *Hits@N* or a lower *Mean Rank* implies the better performance of a model.

TABLE 3. Entity prediction results on FB15k.

Algorithm	MR(Raw)	MR(Filter)	Hits@10(Raw)	Hits@10(Filter)
Unstructured	1074	979	4.5	6.3
RESCAL	828	683	28.4	44.1
SE	273	162	28.8	39.8
LFM	283	164	26.0	33.1
TransE	243	125	34.9	47.1
TransR	198	77	48.2	68.7
CTransR	223	82	44.0	66.7
ProjE	124	34	54.7	88.4
PTransE	200	58	51.4	84.6
TransD	91	89.1	51.8	77.3
DistMult	188.3	86.4	51.9	79.7
ComplEx	-	-	-	84.1
GAKE	228	119	44.5	64.8
ConvE	-	64	-	87.3
R-GCN+	-	75	-	84.2
TCE(H)	160	46	**55.4**	**89.1**
TCE(T)	**110**	**25**	**55.3**	83.1
TCE(TP)	**35**	**24**	**78.6**	**91.8**

2) ENTIY PREDICTION

For this task, we aim to infer the possible h or t entity in a testing triple (h, r, t) when one of them is missing. The results of entity prediction on FB15k are shown in Table 3. For TCE, we consider the different score functions for the conditional probability: TCE(H) uses Hadamard product to compute the connection between triple and its context. In TCE(T), score function of TransE is taken instead of Hadamard product in TCE(H). The special one is TCE(TP) which is TCE(T) model that initializes all embedding by the trained TransE. From the results it can be observed that our model significantly and consistently outperforms most baselines only slightly worse than ProjE on *Mean Rank (Filter and Raw)* and TransD on *Mean Rank (Raw)*. The result implies that leveraging triple context can improve the performance of the embedding model on the entity prediction task. Although GAKE also uses context information for KG embedding, its performance is inferior to the TCE model, which shows the superiority of our embedding mechanism. The reason for the result is GAKE only considers the connection between a single entity or a relation and the context while ignoring the connection in triple itself. The experimental results of ComplEx are absent here, which is because they use a different metric, MRR (Mean Reciprocal rank), instead of *Mean Rank* for evaluation. Nonetheless, according to *Hits@10 (Filter)* reported in [10], the performance of our model are better than ComplEx.

In addition, we also replace the Hadamard Product in our model with the score function of TransE for evaluation. The performance of *Mean Rank* is slightly improved but *Hits@10* is declined. It indicates that other score function can improve the rank of candidate entities, but cannot distinguish between positive candidates and negative candidates well. With a TransE model being trained at first, the embedding of TransE will significantly improve the performance of TCE(T) and obtain the best performance on FB15k.

Table 5 shows the results of entity prediction on FB15k-237. It contains fewer models than Table 3 because some models did not perform the entity prediction task on FB15k-237. For the task, our model with the best performance is chosen. The experimental results on FB15k-237 are similar to the results in FB15k. It can be concluded that:

1) For *Hits@10*, TCE outperforms all other baselines which consider triples independently. Table 6 is the further analysis in FB15k. It can prove the effectiveness of our model because TCE performs better not only on *Hits@10* but also on *Hits@1*, *Hits@3*, and *Hits@5*. Compared to all models, TCE achieves an improvement of 6%/3%/6% on *Hits@1*/*Hits@3*/*Hits@5* respectively.

2) Considering additional structured information, it improves the performance for both *Mean Rank* and *Hits@10*. The reason is that, learning entity h or t with their contexts is similar to ensemble multi DistMult (treated as missing structured information) functions. Take (h_n, r_n, h) and (h, r, t) as example where h_n and r_n is a of h, TCE will optimize the two triples as the same time and can easily distinguish entities with same r and t but with different neighbors.

3) Training TCE with subsets of neighbor contexts and path contexts has already reached comparable results with baseline models. It suggests that it may be unnecessary to take all the contexts into account.

As defined in [8], relations in KGs can be divided into various types according to their mapping types such as 1-to-1, 1-to-N, N-to-1, and N-to-N. Table 4 shows the evaluation results with respect to different types of relations on FB15k. Excluding predicting tail of N-To-1 relations, we can see that TCE(H) and TCE(T) bring promising improvements in

TABLE 4. Entity prediction results on FB15k by mapping types of relations.

Task	Predicting head(HITS@10(%))				Predicting tail(HITS@10(%))			
Relation Category	1-To-1	1-To-N	N-To-1	N-To-N	1-To-1	1-To-N	N-To-1	N-To-N
SE	35.6	62.6	17.2	37.5	34.9	14.6	68.3	41.3
SME(linear)	35.1	53.7	19.0	40.3	32.7	14.9	61.6	43.3
SME(bilinear)	30.9	69.6	19.9	38.6	28.2	13.1	76.0	41.8
TransE	43.7	65.7	18.2	47.2	43.7	19.7	66.7	50.0
TransH (unif)	66.7	81.7	30.2	57.4	63.7	30.1	83.2	60.8
TransH (bern)	66.8	87.6	28.7	64.5	65.5	39.8	83.3	67.2
TransR (unif)	76.9	77.9	38.1	66.9	76.2	38.4	76.2	69.1
TransR (bern)	78.8	89.2	34.1	69.2	79.2	37.4	**90.4**	72.1
CTransR (unif)	78.6	77.8	36.4	68.0	77.4	37.8	78.0	70.3
CTransR (bern)	81.5	89.0	34.7	71.2	80.8	38.6	90.1	73.8
TCE(H)	**90.7**	**90.3**	**92.6**	**89.8**	87.3	**90.5**	**89.9**	83.1
TCE(T)	71.0	60.3	83.9	81.9	70.3	89.9	76.0	**89.2**

TABLE 5. Entity prediction results on FB15k-237.

Algorithm	MR	Hits@10(Filter)
ConvE [10]	330	44.8
IRN	**211**	46.4
TransE	237	35.5
TransD	-	42.7
DistMult	254	33.7
ComplEx	248	42.7
R-GCN+	-	41.7
Neural LP	-	36.7
TCE	363	**48.9**

TABLE 6. Further results of entity prediction on FB15k for Hits@N.

Algorithm	Hits@1	Hits@3	Hits@5
TransE	24.6	49.5	57.6
DistMult	53.2	73.0	76.9
KALE	38.3	61.6	68.3
PTransE	56.5	76.8	81.0
ComplEx	38.3	61.6	68.3
TCE	**62.6**	**79.4**	**87.7**

modeling complex relations. Specifically, TCE(H) performs well when predicting the "N" sides of 1-To-N and N-To-1 relations, which indicates candidate positive triples have higher scores than candidate negative triples in general. In other simpler scenarios, such as 1-To-1 relations and predicting the "1" sides of 1-To-N and N-To-1 relations, the performance of TCE is still better than most baselines. TCE(T) behaves well in predicting tail of 1-To-N relations, predicting head of N-To-1 relations and N-To-N relations. Specifically, on 1-To-1 relations and predicting the "1" side of 1-To-N and N-To-1 relations, the performance of TCE(T) is still acceptable though not so good as some other baselines, such as TransH and TransR.

In a nutshell, Table 4 displays that utilizing triple context for KG embedding is helpful when handling complex relations and improving the precision in modeling simple relations.

3) CASE STUDY

Table 7 shows the triple context improves the performance while predicting the tail entities and relations. The **bold** show correct answers. In the first two examples show predict the tail entity by given a head entity and a relation. In TransE, the correct answers do not appear in Top3. In contrast with TransE, TCE improves the performance and the correct answers are ranking in the first place by given triple contexts which can help to infer the tail entities. The last case shows the triple context improves the performance while predicting the relation between given a head entity and a tail entity. In the last example, TCE knows the place of birth of a person and the speaking language of the head entity. This neighbor information in triple context can help TCE to find the nationality of a person.

4) RELATION PREDICTION

For this task, we aim to predict relations given two entities. Only FB15K is taken for evaluation because there are no baselines for performing this task in FB15k-237. For a comparative study, TransE, ProjE, DKRL, PRA, ARP, and PTransE as chosen as baseline models for relation prediction. The results of TransE, ProjE, DKRL, and PTransE are borrowed from their original papers. The results of PRA and ARP are based on our implementations. There are three kinds of PTransE with two different parameters. The first parameter indicates the operation (ADD/RNN) for a relation path and the second one is the length (2/3) for a relation path. The evaluation results are shown in Table 8, in which we report *Hits@1* instead of *Hits@10* for comparison because *Hits@10* for all models exceeds 95%. From Table 8 it can be observed that:

1) TCE outperforms all models on *Hits@1*, which justifies the effectiveness of our model.

2) By comparing TCE with other methods, the results show that utilizing structured information has a positive effect on relation prediction.

3) By considering relation paths, PTransE obtains high *Mean Rank* and *Hits@1*. However, not all entities in the test

TABLE 7. Examples of predictions on TCE.

Head Entity	Relation	TCE	TransE	Triple Context
h: $indiana\_state\_university$	location	**the_hoosier_state** terre_haute rhode_island	maryland rhode_island the_constitution_state	institution/campuses location/state_province_region
h: $university\_of\_victoria$	location	**victoria** kurnaby kelowna	kelowna toronto ottawa	institution/campuses location/citytown
Head Entity	Tail Entity	TCE	TransE	Triple Context
h: $Kaneto\_Shiozawa$	t: $Japanese$	**Nationality** main_country state_province_region	main_country profession state_province_region	language location/birth_place

TABLE 8. Relation prediction on FB15k.

Algorithm	MR(Raw)	MR(Filter)	Hits@1(Raw)	Hits@1(Filter)
TransE	2.8	2.5	65.1	84.3
DKRL(CNN)	2.6	2.3	67.1	86.7
PTransE(ADD,2)	1.7	1.2	69.5	93.6
PTransE(RNN,2)	1.9	1.4	68.3	93.2
PTransE(ADD,3)	1.8	1.4	68.5	94.0
ProjE	**1.5**	**1.2**	75.5	95.6
PRA	2.5	2.1	70.2	91.6
ARE	6.7	6.3	62.6	83.9
TCE	2.4	1.9	**75.7**	**95.7**

set have relation paths, which may lead to random guesses. Against this, the neighbor context in TCE can be considered as a constraint which is helpful for predicting the relationship between an entity pair with no path context.

VI. CONCLUSION

In this paper, we introduced TCE which is a novel model for the KG embedding task. For each triple, TCE can leverage its triple context which is composed of neighbor context and path context. TCE learns the better embeddings of entities and relations in their triple context. The novel model is evaluated on two benchmark datasets for link prediction and relation prediction. The experiments prove TCE not only can embeddings of a given KG, but also can capture the complex structure in a KG. The performance of the two tasks verifies further that TCE makes significant improvements over the major baselines.

The subsequent research will consider two ways to improve our work: (1) Current results show complementarity to some other methods such as TransH, TransR. A combination of previous models and our model can be considered. (2) The experimental results reveal that not all the neighbors and paths give useful information. Therefore, how to select useful neighbors and paths will be focused on.

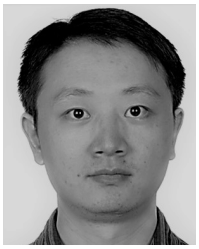
REFERENCES

[1] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph and text jointly embedding," in *Proc. EMNLP*, Doha, Qatar, 2014, pp. 1591–1601.
[2] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. NIPS*, Lake Tahoe, NV, USA, 2013, pp. 3111–3119.
[3] S. Guo, Q. Wang, L. Wang, B. Wang, and L. Guo, "Jointly embedding knowledge graphs and logical rules," in *Proc. EMNLP*, Austin, TX, USA, 2016, pp. 192–202.
[4] N. Lao, T. Mitchell, and W. W. Cohen, "Random walk inference and learning in a large scale knowledge base," in *Proc. EMNLP*, Edinburgh, U.K., 2011, pp. 529–539.
[5] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proc. ICML*, Bellevue, WA, USA, 2011, pp. 809–816.
[6] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Proc. NIPS*, Lake Tahoe, NV, USA, 2013, pp. 926–934.
[7] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. NIPS*, Lake Tahoe, NV, USA, 2013, pp. 2787–2795.
[8] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. AAAI*, Quebec City, QC, Canada, 2014, pp. 1112–1119.
[9] J. Xu, X. Qiu, K. Chen, and X. Huang, "Knowledge graph representation with jointly structural and textual," in *Proc. IJCAI*, Melbourne, VIC, Australia, 2017, pp. 1318–1324.
[10] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2D knowledge graph embeddings," in *Proc. AAAI*, New Orleans, LA, USA, 2018, pp. 1811–1818.
[11] J. Feng, M. Huang, Y. Yang, and X. Zhu, "GAKE: Graph aware knowledge embedding," in *Proc. COLING*, Osaka, Japan, 2016, pp. 641–651.
[12] Y. Lin, Z. Liu, H. Luan, M. Sun, S. Rao, and S. Liu, "Modeling relation paths for representation learning of knowledge bases," in *Proc. EMNLP*, Lisbon, Portugal, 2015, pp. 705–714.
[13] A. Bordes, J. Weston, R. Collobert, and Y. Bengio, "Learning structured embeddings of knowledge bases," in *Proc. AAAI*, San Francisco, CA, USA, 2011, pp. 1–6.
[14] 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. ACL*, Berlin, Germany, 2016, pp. 1–11.
[15] J. Weston, A. Bordes, O. Yakhnenko, and N. Usunier, "Connecting language and knowledge bases with embedding models for relation extraction," in *Proc. EMNLP*, Seattle, WA, USA, 2013, pp. 1366–1371.
[16] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proc. AAAI*, Austin, TX, USA, 2015, pp. 1366–1371.
[17] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *Proc. ICML*, New York, NY, USA, 2016, pp. 2071–2080.
[18] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," in *Proc. ACL*, Beijing, China, 2015, pp. 687–696.
[19] B. Yang, W.-T. Yih, X. He, J. Gao, and L. Deng. (Dec. 2014). "Embedding entities and relations for learning and inference in knowledge bases." [Online]. Available: https://arxiv.org/abs/1412.6575
[20] B. Shi and T. Weninger, "ProjE: Embedding projection for knowledge graph completion," in *Proc. AAAI*, San Francisco, CA, USA, 2017, pp. 1236–1242.
[21] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon, "Representing text for joint embedding of text and knowledge bases," in *Proc. EMNLP*, Lisbon, Portugal, 2015, pp. 1499–1509.
[22] T. Trouillon, C. R. Dance, É. Gaussier, J. Welbl, S. Riedel, and G. Bouchard, "Knowledge graph completion via complex tensor factorization," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 130:1–130:38, 2017.

[23] D. Krompaß, S. Baier, and V. Tresp, "Type-constrained representation learning in knowledge graphs," in *Proc. ISWC*, Bethlehem, PA, USA, 2015, pp. 640–655.
[24] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. SIGKDD*, New York, NY, USA, 2014, pp. 701–710.
[25] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. WWW*, Florence, Italy, 2015, pp. 1067–1077
[26] C. Tu, H. Liu, Z. Liu, and M. Sun, "CANE: Context-aware network embedding for relation modeling," in *Proc. ACL*, Vancouver, BC, Canada, 2017, pp. 1722–1731.
[27] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "DBpedia: A nucleus for a Web of open data," in *Proc. ISWC*, Busan, South Korea, 2007, pp. 722–735.
[28] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: A core of semantic knowledge," in *Proc. WWW*, Banff, AB, Canada, 2007, pp. 697–706.
[29] T. Mitchell *et al.*, "Never-ending learning," in *Proc. AAAI*, Austin, TX, USA, 2015, pp. 2302–2310.
[30] J. Shi, H. Gao, G. Qi, and Z. Zhou, "Knowledge graph embedding with triple context," in *Proc. CIKM*, Singapore, 2017, pp. 2299–2302.
[31] X. Dong *et al.*, "Knowledge vault: A Web-scale approach to probabilistic knowledge fusion," in *Proc. SIGKDD*, New York, NY, USA, 2014, pp. 601–610.
[32] M. Nickel, X. Jiang, and V. Tresp, "Reducing the rank in relational factorization models by including observable patterns," in *Proc. NIPS*, Quebec City, QC, Canada, 2014, pp. 1179–1187.
[33] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. ESWC*, Heraklion, Greece, 2018, pp. 593–607.

HUAN GAO received the B.S. degree in computer science and technology from Xi'an Shiyou University, Xi'an, China, in 2008, and the M.S. degree from Shanghai University in 2013. He is currently pursuing the Ph.D. degree in software engineering at Southeast University. His current research interests include knowledge graphs, data mining, and natural language processing.

JUN SHI received the B.S. degree in computer science and engineering from Southeast University, Nanjing, China, in 2015, where he is currently pursuing the M.S. degree in software engineering. His research interests include knowledge graphs and natural language processing.



GUILIN QI received the Ph.D. degree in computer science from Queen's University Belfast in 2006. He was with the Institute AIFB, University of Karlsruhe, for three years. He is currently a Professor at Southeast University, China, where he is also the Head of the Knowledge Science and Engineering Lab and the Director of the Institute of Cognitive Science. He has published over 100 papers in these areas, many of which were published in proceedings of major conferences or journals. He is a Work Package Leader of an EU FP7 Marie Curie IRSES Project and a Co-Investigator of an ARC Discovery Project. He has published a book on knowledge management for the semantic Web in 2015. His research interests include knowledge representation and reasoning, knowledge graph, uncertainty reasoning, and semantic Web. He received the Best Short Paper Runner-Up Award from CIKM 2017 and has a paper received the Best-Student Paper Award in ICTAI 2015. He is an Executive Editor-in-Chief of Data Intelligence and an Associate Editor of the *Journal of Web Semantics*.

MENG WANG received the B.S. degree in computing science from Sichuan University in 2012 and the Ph.D. degree from Xi'an Jiaotong University. His research interests include knowledge graph, network embedding, SPARQL query language, and data mining.