Orthogonal Relation Transforms with Graph Context Modeling for Knowledge Graph Embedding

Yun Tang, Jing Huang, Guangtao Wang, Xiaodong He, Bowen Zhou
JD AI Research
{yun.tang,jing.huang,guangtao.wang,xiaodong.he,bowen.zhou}@jd.com

Abstract

Distance-based knowledge graph embeddings have shown substantial improvement on the knowledge graph link prediction task, from *TransE* to the latest state-of-the-art *RotatE*. However, complex relations such as N-to-1, 1-to-N and N-to-N still remain challenging to predict. In this work, we propose a novel distance-based approach for knowledge graph link prediction. First we extend the *RotatE* from 2D complex domain to high dimensional space with orthogonal transforms to model relations. The orthogonal transform embedding for relations keeps the capability for modeling symmetric/anti-symmetric, inverse and compositional relations while achieves better modeling capacity. Second, the graph context is integrated into distance scoring functions directly. Specifically, graph context is explicitly modeled via two directed context representations. Each node embedding in knowledge graph is augmented with two context representations, which are computed from the neighboring outgoing and incoming nodes/edges respectively. The proposed approach improves prediction accuracy on the difficult N-to-1, 1-to-N and N-to-N cases. Our experimental results show that it achieves state-of-the-art results on two common benchmarks FB15k-237 and WNRR-18, especially on FB15k-237 which has many high in-degree nodes. Code available at https://github.com/JD-AI-Research-Silicon-Valley/KGEmbedding-OTE.

1 Introduction

Knowledge graph is a multi-relational graph whose nodes represent entities and edges denote relationships between entities. Knowledge graphs store facts about people, places and world from various sources. Those facts are kept as triples (head entity, relation, tail entity) and denoted as (h, r, t). A large number of knowledge graphs, such as Freebase (Bollacker et al., 2008), DBpedia (Auer et al., 2007), NELL (Carlson et al., 2010) and YAGO3 (Mahdisoltani et al., 2013), have been built over the years and successfully applied to many domains such as recommendation and question answering (Bordes et al., 2014; Zhang et al., 2016). However, these knowledge graphs need to be updated with new facts periodically. Therefore many knowledge graph embedding methods have been proposed for link prediction that is used for knowledge graph completion.

Knowledge graph embedding represents entities and relations in continuous vector spaces. Started from a simple and effective approach called *TransE* (Bordes et al., 2013), many knowledge graph embedding methods have been proposed, such as *TransH* (Wang et al., 2014), *DistMult* (Yang et al., 2014), *ConvE* (Dettmers et al., 2018) to the latest *RotatE* (Sun et al., 2019) and *QuatE* (Zhang et al., 2019).

Though much progress has been made, 1-to-N, N-to-1, and N-to-N relation predictions (Bordes et al., 2013; Wang et al., 2014) still remain challenging. In Figure 1, relation "profession" demonstrates an N-to-N example and the corresponding edges are highlighted as green. Assuming the triple (SergeiRachmaninoff, Profession, Pianist) is unknown. The link prediction model takes "SergeiRachmaninoff" and relation "Profession" and rank all entities in the knowledge graph to predict "Pianist". Entity "SergeiRachmaninoff" connected to multiple entities as head entity via relation "profession", while "Pianist" as a tail entity also reaches to multiple entities through relation "profession". It makes the N-to-N prediction hard because the mapping from certain entity-relation pair could lead to multiple different entities. Same issue happens with the case of 1-to-N and N-to-1 predictions.

The recently proposed *RotatE* (Sun et al., 2019)

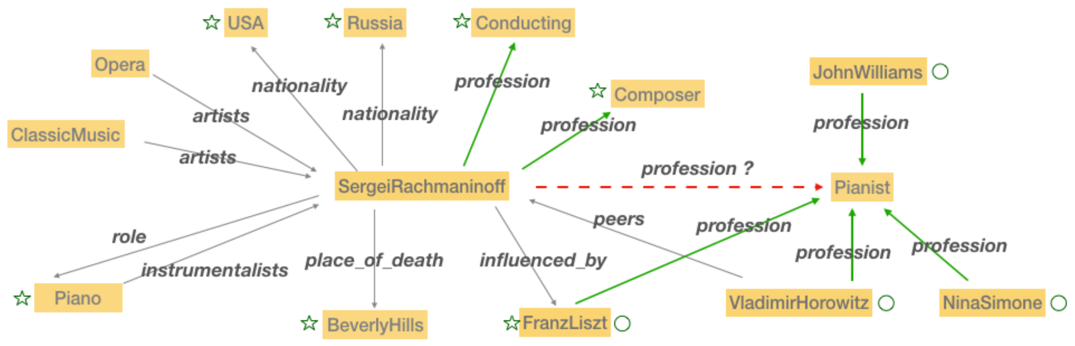

Figure 1: Snapshot of knowledge graph in FB15k-237. Entities are represented as golden blocks.

models each relation as a 2-D rotation from the source entity to the target entity. The desired properties for relations include symmetry/antisymmery, inversion and composition which have been demonstrated to be useful for link prediction in knowledge graph. Many existing methods model one or a few of these relation patterns, while *RotatE* naturally handles all these relation patterns. In addition, the entity and relation embeddings are divided into multiple groups (for example, 1000 2-D rotations are used in (Sun et al., 2019)). Each group is modeled and scored independently. The final score is computed as the summation of all these scores, which can be viewed as an ensemble of different models and further boost the performance of link prediction. However, *RotatE* is limited to 2-D rotations and thus has limited modeling capacity. In addition, *RotatE* does not consider graph context, which is helpful in handling 1-to-N, N-to-1, and N-to-N relation prediction.

In this work, a novel distance-based knowledge graph embedding called orthogonal transform embedding (*OTE*) with graph context is proposed to alleviate the 1-to-N, N-to-1 and N-to-N issues, while keeps the desired relation patterns as *RotatE*. First, we employ orthogonal transforms to represent relations in high dimensional space for better modeling capability. The Orthogonal transform embedding also models the symmetry/antisymmery, inversion and compositional relation patterns just as *RotatE* does. *RotatE* can be viewed as an orthogonal transform in 2D complex space.

Second, we integrate graph context directly into the distance scoring, which is helpful to predict 1-to-N, N-to-1 and N-to-N relations. For example, from the incomplete knowledge graph, people find useful context information, such as (SergeiRachmaninoff, role, Piano) and (SergeiRachmaninoff, Profession, Composer) in Figure 1. In this work, each node embedding in knowledge graph is augmented with two graph context representations, computed from the neighboring outgoing and incoming nodes respectively. Each context representation is computed based on the embeddings of the neighbouring nodes and the corresponding relations connecting to these neighbouring nodes. These context representations are used as part of the distance scoring function to measure the plausibility of the triples during training and inference. We show that *OTE* together with graph context modeling performs consistently better than *RotatE* on the standard benchmark FB15k-237 and WN18RR datasets.

In summary, our main contributions include:

- A new orthogonal transform embedding *OTE*, is proposed to extend *RotatE* from 2D space to high dimensional space, which also models symmetry/antisymmery, inversion and compositional relation patterns;
- A directed graph context modeling method is proposed to integrate knowledge graph context (including both neighboring entity nodes and relation edges) into the distance scoring function;
- Experimental results of *OTE* on standard benchmark FB15k-237 and WN18RR datasets show consistent improvements over *RotatE*, the state of art distance-based embedding model, especially on FB15k-237 with many high in-degree nodes. On WN18RR our results achieve the new state-of-the-art performance.

2 Related work

2.1 Knowledge Graph Embedding

Knowledge graph embedding could be roughly categorized into two classes (Wang et al., 2017): distance-based models and semantic matching models. Distance-based model is also known as additive models, since it projects head and tail entities into the same embedding space and the distance scoring between two entity embeddings is used to measure the plausibility of the given triple. *TransE* (Bordes et al., 2013) is the first and most representative translational distance model. A series of work is conducted along this line such as *TransH* (Wang et al., 2014), *TransR* (Lin et al., 2015) and *TransD* (Ji et al., 2015) etc. *RotatE* (Sun et al., 2019) further extends the computation into complex domain and is currently the state-of-art in this category. On the other hand, Semantic matching models usually take multiplicative score functions to compute the plausibility of the given triple, such as *DistMult* (Yang et al., 2014), *ComplEx* (Trouillon et al., 2016), *ConvE* (Dettmers et al., 2018), *TuckER* (Balazevic et al., 2019) and *QuatE* (Zhang et al., 2019). *ConvKB* (Nguyen et al., 2017) and *CapsE* (Nguyen et al., 2019) further took the triple as a whole, and fed head, relation and tail embeddings into convolutional models or capsule networks.

The above knowledge graph embedding methods focused on modeling individual triples. However, they ignored knowledge graph structure and did not take advantage of context from neighbouring nodes and edges. This issue inspired the usage of graph neural networks (Kipf and Welling, 2016; Veličković et al., 2017) for graph context modeling. Encoder-decoder framework was adopted in (Schlichtkrull et al., 2017; Shang et al., 2019; Bansal et al., 2019). The knowledge graph structure is first encoded via graph neural networks and the output with rich structure information is passed to the following graph embedding model for prediction. The graph model and the scoring model could be end-to-end trained together, or the graph encoder output was only used to initialize the entity embedding (Nathani et al., 2019). We take another approach in this paper: we integrate the graph context directly into the distance scoring function.

2.2 Orthogonal Transform

Orthogonal transform is considered to be more stable and efficient for neural networks (Saxe et al., 2013; Vorontsov et al., 2017). However, to optimize a linear transform with orthogonal property reserved is not straightforward. Soft constraints could be enforced during optimization to encourage the learnt linear transform close to be orthogonal. Bansal et al. (2018) extensively compared different orthogonal regularizations and find regularizations make the training faster and more stable in different tasks. On the other hand, some work has been done to achieve strict orthogonal during optimization by applying special gradient update scheme. Harandi and Fernando (2016) proposed a Stiefel layer to guarantee fully connected layers to be orthogonal by using Reimannian gradients. Huang et al. (2017) consider the estimation of orthogonal matrix as an optimization over multiple dependent stiefel manifolds problem and solve it via eigenvalue decomposition on a proxy parameter matrix. Vorontsov et al. (2017) applied hard constraint on orthogonal transform update via Cayley transform. In this work, we construct the orthogonal matrix via Gram Schmidt process and the gradient is calculated automatically through autograd mechanism in PyTorch (Paszke et al., 2017).

3 Our Proposed Method

We consider knowledge graph as a collection of triples $\mathcal{D} = \{(h, r, t)\}$ with V as the graph node set, and R as the graph edge set. Each triple has a head entity h and tail entity t, where $h, t \in V$. Relation $r \in R$ connects two entities with direction from head to tail. As discussed in the introduction section, 1-to-N, N-to-1 and N-to-N relation prediction (Bordes et al., 2013; Wang et al., 2014) are difficult to deal with. They are addressed in our proposed approach by: 1) orthogonal relation transforms that operate on groups of embedding space. Each group is modeled and scored independently, and the final score is the sum of all group scores. Hence, each group could address different aspects of entity-relation pair and alleviate the 1-to-N and N-to-N relation mapping issues; and 2) directed graph context to integrate knowledge graph structure information to reduce the ambiguity.

Next, we first briefly review *RotatE* that motivates our orthogonal transform embedding (*OTE*), and then describe the proposed method in details.

3.1 RotatE

OTE is inspired by *RotatE* (Sun et al., 2019). In *RotatE*, the distance scoring is done via Hadamard production (element-wise) defined on the complex domain. Given a triple (h, r, t), the corresponding embedding are e_h, θ_r, e_t, where e_h and $e_t \in \mathcal{R}^{2d}$, $\theta_r \in \mathcal{R}^{d}$, and d is the embedding dimension. For each dimension i, $e[2i]$ and $e[2i+1]$ are corresponding real and imaginary components. The projection $\tilde{e}_t$ of t from corresponding relation and head entities is conducted as an orthogonal transform as below:

$$\begin{bmatrix} \tilde{e}_t[2i] \\ \tilde{e}_t[2i+1] \end{bmatrix} = M_r(i) \begin{bmatrix} e_h[2i] \\ e_h[2i+1] \end{bmatrix}$$
$$= \begin{bmatrix} \cos\theta_r(i) & -\sin\theta_r(i) \\ \sin\theta_r(i) & \cos\theta_r(i) \end{bmatrix} \begin{bmatrix} e_h[2i] \\ e_h[2i+1] \end{bmatrix}$$

where $M_r(i)$ is a 2D orthogonal matrix derived from θ_r .

Though *RotatE* is simple and effective for knowledge graph link prediction, it is defined in $2D$ complex domain and thus has limited modeling capability. A natural extension is to apply similar operation on a higher dimensional space.

3.2 Orthogonal Transform Embedding (OTE)

We use e_h, M_r, e_t to represent embeddings of head, relation and tail entity, where e_h, $e_t \in \mathcal{R}^d$, and d is the dimension of the entity embedding. The entity embedding e_x, where $x = \{h, t\}$, is further divided into K sub-embeddings, e.g., $e_x = [e_x(1); \cdots; e_x(K)]$, where $e_x(i) \in \mathcal{R}^{d_s}$ and $d = K \cdot d_s$. M_r is a collection of K linear transform matrix $M_r = \{M_r(1), \cdots, M_r(K)\}$, and $M_r(i) \in \mathcal{R}^{d_s \times d_s}$.

For each sub-embedding $e_t(i)$ of tail t, we define the projection from h and r to t as below:

$$\tilde{e}_t(i) = f_i(h, r) = \phi(M_r(i))e_h(i) \quad (1)$$

where ϕ is the Gram Schmidt process (see details in Section 3.3) applied to square matrix $M_r(i)$. The output transform $\phi(M_r(i))$ is an orthogonal matrix derived from $M_r(i)$. $\tilde{e}_t$ is the concatenation of all sub-vector $\tilde{e}_t(i)$ from Eq. 1, e.g., $\tilde{e}_t = f(h, r) = [\tilde{e}_t(1); \cdots; \tilde{e}_t(K)]$. The L_2 norm of $e_h(i)$ is preserved after the orthogonal transform. We further use a scalar tensor $s_r(i) \in \mathcal{R}^{d_s}$ to scale the L_2 norm of each group of embedding separately. Eq. 1 is re-written as

$$\tilde{e}_t(i) = diag(\exp(s_r(i)))\phi(M_r(i))e_h(i) \quad (2)$$

Then, the corresponding distance scoring function is defined as

$$d((h, r), t) = \sum_{i=1}^{K}(||\tilde{e}_t(i) - e_t(i)||) \quad (3)$$

For each sub-embedding $e_h(i)$ of head h, we define the projection from r and t to h as below:

$$\tilde{e}_h(i) = diag(\exp(-s_r(i)))\phi(M_r(i))^T e_t(i) \quad (4)$$

where the reverse project from tail to head is simply transposing the $\phi(M_r(i))$ and reversing the sign of s_r. Then, the corresponding distance scoring function is defined as

$$d(h, (r, t)) = \sum_{i=1}^{K}(||\tilde{e}_h(i) - e_h(i)||). \quad (5)$$

3.3 Gram Schmidt Process

We employ Gram-Schmidt process to orthogonalize a linear transform into an orthogonal transform (i.e., $\phi(M_r(i))$ in Section 3.2). The Gram-Schmidt process takes a set of tensor $S = \{v_1, \cdots, v_k\}$ for $k \leq d_s$ and generates an orthogonal set $S' = \{u_1, \cdots, u_k\}$ that spans the same $k-$dimensional subspace of $\mathcal{R}^{d_s}$ as S.

$$t_i = v_k - \sum_{j=1}^{k-1} \frac{\langle v_k, t_j \rangle}{\langle t_j, t_j \rangle} t_j \quad (6)$$
$$u_i = \frac{t_i}{||t_i||} \quad (7)$$

where $t_1 = v_1$, $||t||$ is the L_2 norm of vector t and $\langle v, t \rangle$ denotes the inner product of v and t.

Orthogonal transform has many desired properties, for example, the inverse matrix is obtained by simply transposing itself. It also preserves the L_2 norm of a vector after the transform. For our work, we are just interested in its property to obtain inverse matrix by simple transposing. This saves the number of model parameters (see Table 3).

It can be easily proved that *OTE* has the ability to model and infer all three types of relation patterns: symmetry/antisymmetry, inversion, and composition as *RotatE* does. The proof is listed in Appendix A.

It should be noted that, $M_r(i)$ is calculated every time in the neural networks forward computation to get orthogonal matrix $\phi(M_r(i))$, while the corresponding gradient is calculated and propagated back to $M_r(i)$ via autograd computation within PyTorch during the backward computation. It eliminates the need of special gradient update schemes employed in previous hard constraint based orthogonal transform estimations (Harandi and Fernando, 2016; Vorontsov et al., 2017). In our experiments, we initialize $M_r(i)$ to make sure they are with full rank[1]. During training, we also keep checking the determinant of $M_r(i)$. We find the update is fairly

[1]A real random matrix has full rank with probability 1 (Slinko, 2000). We use different random seeds to make sure the generated matrix is full rank.

stable that we don't observe any issues with sub-embedding dimensions varied from 5 to 100.

3.4 Directed Graph Context

The knowledge graph is a directed graph: valid triple (h, r, t) does not mean (t, r, h) is also valid. Therefore, for a given entity in knowledge graph, there are two kinds of context information: nodes that come into it and nodes that go out of it. Specially, in our paper, for each entity e, we consider the following two context settings:

1. If e is a tail, all the (head, relation) pairs in the training triples whose tail is e are defined as *Head Relation Pair Context*.
2. If e is a head, all the (relation, tail) pairs in the training triples whose head is e are defined as *Relation Tail Pair Context*.

Figure 1 demonstrates the computation of graph context for a testing triple (SergeiRachmaninoff, profession, Pianist). Edges for relation "profession" are colored as green. Entities marked with ○ are head entities to entity "Pianist", and these entities and corresponding relations to connect "Pianist" form the head relation pair context of "Pianist". While entities with ☆ are tail entities for entity "SergeiRachmaninoff". Those entities and corresponding relations are the relation tail graph context of entity "SergeiRachmaninoff".

3.4.1 Head Relation Pair Context

For a given tail t, all head-relation pairs (h', r') of the triples with tail as t are considered as its graph context and denoted as $Ng(t)$.

First, we compute the head-relation context representation $\tilde{e}_t^c$ as the average from all these pairs in $Ng(t)$ as below:

$$\tilde{e}_t^c = \frac{\sum_{(h',r')\in Ng(t)} f(h', r') + e_t}{|Ng(t)| + 1} \tag{8}$$

where e_t is the embedding of the tail t, $f(h', r')$ is the representation of (h', r') induced from Eq. 2. We use e_t in Eq. 8 to make the computation of context representation possible when $Ng(t)$ is empty. This can be viewed as a kind of additive smoothing for context representation computation.

Then, we compute the distance of the head-relation context of t and the corresponding orthogonal transform based representation of a triple (h, r, t) as follow.

$$d_c((h, r), t) = \sum_{i=1}^{K} (||\tilde{e}_t(i) - \tilde{e}_t^c(i)||) \tag{9}$$

where $\tilde{e}_t(i)$ is computed from Eq. 2.

There is no new parameter introduced for the graph context modeling, since the message passing is done via *OTE* entity-relation project $f(h', r')$. The graph context can be easily applied to other translational embedding algorithms, such as *RotatE* and *TransE* etc, by replacing *OTE*.

3.4.2 Relation Tail Pair Context

For a given head h, all relation-tail pairs (r', t') of the triples with head as h are considered as its graph context and denoted as $Ng(h)$.

First, we compute the relation-tail context representation $\tilde{e}_h^c$ as the average from all these pairs in $Ng(h)$ as below:

$$\tilde{e}_h^c = \frac{\sum_{(r',t')\in Ng(h)} f(r', t') + e_h}{|Ng(h)| + 1} \tag{10}$$

where $f(r', t')$ is computed from Eq. 4.

Then, we compute the distance of the relation-tail context of h and the corresponding orthogonal transform based representation of a triple (h, r, t) as follow.

$$d_c(h, (r, t)) = \sum_{i=1}^{K} (||\tilde{e}_h(i) - \tilde{e}_h^c(i)||) \tag{11}$$

where $\tilde{e}_h(i)$ is computed from Eq. 4.

3.5 Scoring Function

We further combine all four distance scores (Eq. 3, Eq. 5, Eq. 9 and Eq. 11) discussed above as the final distance score of the graph contextual orthogonal transform embedding (*GC-OTE*) for training and inference

$$\begin{aligned} d_{all}(h, r, t) &= d((h, r), t) + d_c(h, (r, t)) \\ &+ d(h, (r, t)) + d_c((h, r), t). \end{aligned} \tag{12}$$

Therefore the full GC-OTE model can be seen as an ensemble of K local GC-OTE models. This view provides an intuitive explanation for the success of GC-OTE.

Optimization Self-adversarial negative sampling loss (Sun et al., 2019) is used to optimize the embedding in this work,

$$\begin{aligned} L \;=\; & -\sum p(h', r, t') \log \sigma(d_{all}(h', r, t') - \gamma) \\ & - \log \sigma(\gamma - d_{all}(h, r, t)) \end{aligned} \tag{13}$$

where γ is a fixed margin, σ is sigmoid function, (h', r, t') is negative triple, and $p(h', r, t')$ is the negative sampling weight defined in (Sun et al., 2019).

4 Experiments

4.1 Datasets

Two commonly used benchmark datasets (FB15k-237 and WN18RR) are employed in this study to evaluate the performance of link prediction.

FB15k-237 (Toutanova and Chen, 2015) dataset contains knowledge base relation triples and textual mentions of Freebase entity pairs. The knowledge base triples are a subset of the FB15K (Bordes et al., 2013), originally derived from Freebase. The inverse relations are removed in FB15k-237.

WN18RR (Dettmers et al., 2018) is derived from WN18 (Bordes et al., 2013), which is a subset of WordNet. WN18 consists of 18 relations and 40,943 entities. However, many text triples obtained by inverting triples from the training set. Thus WN18RR (Dettmers et al., 2018) is created to ensure that the evaluation dataset does not have test leakage due to redundant inverse relation.

Dataset	FB15k-237	WN18RR
Entities	14,541	40,943
Relations	237	11
Train Edges	272,115	86,835
Val. Edges	17,535	3,034
Test Edges	20,466	3,134

Table 1: Statistics of datasets.

Each dataset is split into three sets for: training, validation and testing, which is same with the setting of (Sun et al., 2019). The statistics of two data sets are summarized at Table 1. Only triples in the training set are used to compute graph context.

4.2 Evaluation Protocol

Following the evaluation protocol in (Dettmers et al., 2018; Sun et al., 2019), each test triple (h, r, t) is measured under two scenarios: head focused $(?, r, t)$ and tail focused $(h, r, ?)$. For each case, the test triple is ranked among all triples with masked entity replaced by entities in knowledge graph. Those true triples observed in either train/validation/test set except the test triple will be excluded during evaluation. Top 1, 3, 10 (Hits@1, Hits@3 and Hits@10), and the Mean Reciprocal Rank (MRR) are reported in the experiments.

4.3 Experimental Setup

Hyper-parameter settings The hyper-parameters of our model are tuned by grid search during training process, including learning rate, embedding dimension d and sub-embedding dimension d_s. In our setting, the embedding dimension is defined as the number of parameters in each entity embedding. Each entity embedding consists of K sub-embeddings with dimension d_s, i.e., $d = K \times d_s$. There are two steps in our model training: 1) the model is trained with *OTE* or *RotatE* models, and 2) graph context based models are fine tuned on these pre-trained models. The parameter settings are selected by the highest MRR with early stopping on the validation set. We use the adaptive moment (Adam) algorithm (Kingma and Ba, 2014) to train the models.

Specially, for FB15k-237, we set embedding dimension $d = 400$, sub-embedding dimension $d_s = 20$, and the learning rates to $2e$-3 and $2e$-4 for pre-training and fine-tuning stages respectively; for WN18RR dataset, we set $d = 400$, $d_s = 4$, and the learning rates to $1e$-4 and $3e$-5 for pre-training and fine-tuning stages.

Implementation Our models are implemented by PyTorch and run on NVIDIA Tesla P40 Graphics Processing Units. The pre-training *OTE* takes 5 hours with 240,000 steps and fine-tuning *GC-OTE* takes 23 hours with 60,000 steps. Though, it takes more computation for graph context based model training, the inference could be efficient if both head and tail context representations are pre-computed and saved for each entity in the knowledge graph.

4.4 Experimental Results

In this section, we first present the results of link prediction, followed by the ablation study and error analysis of our models.

4.4.1 Results of Link Prediction

Table 2 compares the proposed models (*OTE* and graph context based *GC-OTE*) to several state-of-the-art models: including translational distance based *TransE* (Bordes et al., 2013), *RotatE* (Sun et al., 2019); semantic matching based *DistMult* (Yang et al., 2014), *ComplEx* (Trouillon et al., 2016), *ConvE* (Dettmers et al., 2018), *TuckER* (Balazevic et al., 2019) and *QuatE* (Zhang et al., 2019), and graph context information based *R-GCN+* (Schlichtkrull et al., 2017), *SACN* (Shang et al., 2019) and A2N (Bansal et al., 2019). These

Model	FB15k-237				WN18RR			
	MRR	H1	H3	H10	MRR	H1	H3	H10
TransE	.294	-	-	.465	.226	-	-	.501
RotatE	.338	.241	.375	.533	.476	.428	.492	.571
DistMult	.241	.155	.263	.419	.43	.39	.44	.49
ComplEx	.247	.158	.275	.428	.44	.41	.46	.51
ConvE	.325	.237	.356	.501	.43	.40	.44	.52
QuatE	.348	.248	.382	.550	.488	.438	.508	.582
TurkER	.358	.266	.392	.544	.470	.443	.482	.526
R-GCN+	.249	.151	.264	.417	-	-	-	-
SACN	.352	.261	.385	.536	.47	.43	.48	.54
A2N	.317	.232	.348	.486	.45	.42	.46	.51
OTE	.351	.258	.388	.537	.485	.437	.502	.587
GC-OTE	**.361**	**.267**	**.396**	**.550**	**.491**	**.442**	**.511**	**.583**

Table 2: Link prediction for FB15k-237 and WN18RR on test sets.

baseline numbers are quoted directly from published papers.

From Table 2, we observe that: 1) on FB15k-237, *OTE* outperforms *RotatE*, and *GC-OTE* outperforms all other models on all metrics. Specifically MRR is improved from 0.338 in *RotatE*, to 0.361, about 7% relative performance improvement. *OTE* which increases sub-embedding dimension from 2 to 20, and graph context each contributes about half the improvement; 2) on WN18RR, *OTE* outperforms *RotatE* and *GC-OTE* achieves the new state-of-the-art results (as far as we know from published papers). These results show the effectiveness of the proposed *OTE* and graph context for the task of predicting missing links in knowledge graph.

Moreover, *GC-OTE* improves more on FB15k-237 than on WN18RR. This is because FB15k-237 has richer graph structure context compared to WN18RR: an average of 19 edges per node v.s. 2 edges per node in WN18RR. These results indicate that the proposed method *GC-OTE* is more effective on data set with rich context structure information.

Model	d_s	MRR	@10	#param
RotatE-S	-	.330	.515	5.9
RotatE-L	-	.340	.530	29.3
OTE	2	.327	.511	6.1
OTE	20	.355	.540	7.8
OTE - scalar	20	.352	.535	7.7
LNE	20	.354	.538	9.6
GC-RotatE-L	-	.354	.546	29.3
GC-OTE	20	.367	.555	7.8

Table 3: Ablation study on FB15k-237 validation set.

4.4.2 Ablation Study

Table 3 shows the results of ablation study of the proposed models and compares the number of model parameters with *RotatE* on FB15k-237 validation set. We perform the ablation study with embedding dimension of 400. The entity embedding dimension for *RotatE-S* and *RotatE-L* are 400 and 2000, respectively.

First we notice that increasing embedding size from 400 to 2000 makes *RotatE* model size more than quadrupled while the performance gain is very limited (Row 1 and 2 in Table 3); increasing group embedding size from 2 to 20 does not increase the model size of *OTE* much, but with nice performance gain (Row 3 and 4 in Table 3). The model size of *OTE* is less than one-third of the size of *RotatE-L* but with better performance. This shows the effectiveness of the *OTE*.

We examine the proposed model in terms of the following aspects:

Impact of sub-embedding dimension: we fix the embedding dimension as 400, and increase the sub-embedding dimension d_s from 2 to 20, the MRR of *OTE* is improved from 0.327 to 0.355 (See Row 3 and Row 4). For *RotatE*, the entity is embedded in complex vector space, this is similar to our setting with sub-embedding dimension = 2. Our results show that increasing the sub-dimension with *OTE* is beneficial to link prediction.

Impact of orthogonal transform: we replace the orthogonal transform operation in *OTE* with two different settings, 1) removing the diagonal scalar tensor as Eq. 1 (See *OTE-scalar*) and 2) using normal linear transform rather than orthogonal transform (See *LNE*). Both settings lead to MRR degradation. This indicates the proposed orthogonal transform is effective in modeling the relation patterns which are helpful for link prediction.

Impact of graph context: we add the graph context based model to both *OTE* (See *GC-OTE*) and *RotatE-L* (See *GC-RotatE-L*). We observe that MRRs are improved for both *RotatE-L* and *OTE*. This shows the importance of modeling context information for the task of link prediction.

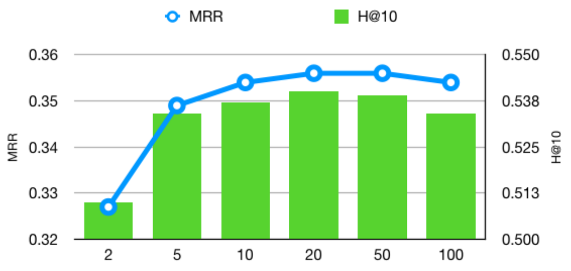

Figure 2: FB15k-237 for *OTE* with different sub-embedding dimension.

Sub-embedding dimension size: in Table 3 we show that increasing sub-embedding dimension brings a nice improvement on MRR. Is the larger size always better? Figure 2 shows the impact of d_s on the *OTE* performance with the changing of sub-embedding size. We fix the entity embedding dimension as 400, and vary the sub-embedding size from $2, 5, 10, 20, 50$, all the way to 100. The blue line and green bar represent MRR and H@10 value, respectively.

From Figure 2 we observe that, both MRR and Hit@10 are improved and slowly saturated around $d_s = 20$ The similar experiments are also conducted on WN18RR data set and we find the best sub-embedding dimension is 4 on WN18RR.

Type	Num.	RotatE-L			GC-OTE		
		H	T	A	H	T	A
1-to-N	2255	.710	.169	.440	.718	.204	.461
N-to-1	5460	.156	.850	.503	.209	.863	.536
N-to-N	9763	.490	.631	.561	.508	.651	.579

Table 4: H@10 from FB15-237 validation set by categories (1-to-N, N-to-1 and N-to-N).

4.4.3 Error Analysis

We present error analysis of the proposed model on 1-to-N, N-to-1 and N-to-N relation predictions on FB15k-237. Table 4 shows results in terms of Hit@10, where "Num." is the number of triples in the validation set belonging to the corresponding category, "H"/"T" represents the experiment to predict head entity /tail entity, and "A" denotes average result for both "H" and "T".

Assume $c(h, r)$ and $c(r, t)$ are the number of (h, r) and (r, t) pairs appeared in triples from the training set respectively. A triple (h, r, t) from the validation set is considered as one of the categories in the following:

$$(h,r,t) = \begin{cases} \text{N-to-1, if } c(h,r) > 1 \text{ and } c(r,t) \leq 1 \\ \text{1-to-N, if } c(h,r) \leq 1 \text{ and } c(r,t) > 1 \\ \text{N-to-N,if } c(h,r) > 1 \text{ and } c(r,t) > 1 \\ \text{other.} \end{cases}$$

From Table 4 we observe that, comparing to *RotatE* large model, the proposed model get better Hit@10 on all cases, especially for the difficult cases when we attempt to predicting the head entity for 1-to-N/N-to-N relation type, and tail entity in N-to-1/N-to-N relation type. The reason is because that in the proposed model, the groupings of sub-embedding relation pairs in *OTE* and graph context modeling both are helpful to distinguish N different tails/heads when they share the same (head, rel)/(rel, tail).

5 Conclusions

In this paper we propose a new distance-based knowledge graph embedding for link prediction. It includes two-folds. First, *OTE* extends the modeling of *RotatE* from 2D complex domain to high dimensional space with orthogonal relation transforms. Second, graph context is proposed to integrate graph structure information into the distance scoring function to measure the plausibility of the triples during training and inference.

The proposed approach effectively improves prediction accuracy on the difficult N-to-1, 1-to-N and N-to-N link predictions. Experimental results on standard benchmark FB15k-237 and WN18RR show that *OTE* improves consistently over *RotatE*, the state-of-the-art distance-based embedding model, especially on FB15k-237 with many high in-degree nodes. On WN18RR our model achieves the new state-of-the-art results. This work is partially supported by Beijing Academy of Artificial Intelligence (BAAI).

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019. TuckER: Tensor factorization for knowledge graph completion. In *EMNLP*.

Nitin Bansal, Xiaohan Chen, and Zhangyang Wang. 2018. Can we gain more from orthogonality regularizations in training deep networks? In *NeurIPS*.

Trapit Bansal, Da-Cheng Juan, Shrividya Ravi, and Andrew McCallum. 2019. A2N: Attending to neighbors for knowledge graph inference. In *ACL*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question answering with subgraph embeddings. In *EMNLP*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NeurIPS*.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *AAAI*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D knowledge graph embeddings. In *AAAI*.

Mehrtash Harandi and Basura Fernando. 2016. Generalized backpropagation, étude de cas: Orthogonality. *ArXiv*.

Lei Huang, Xianglong Liu, Bo Lang, Adams Wei Yu, Yongliang Wang, and Bao Qin Li. 2017. Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks. In *AAAI*.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *ACL*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *ICLR*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embedings for knowledge graph completion. In *AAAI*.

Farzaneh Mahdisoltani, Joanna Biega, and Fabian M Suchanek. 2013. Yago3: A knowledge base from multilingual wikipedias. In *CIDR*.

Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs. In *ACL*.

Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2017. A novel embedding model for knowledge base completion based on convolutional neural network. *arXiv preprint arXiv:1712.02121*.

Dai Quoc Nguyen, Thanh Vu, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2019. A Capsule Network-based Embedding Model for Knowledge Graph Completion and Search Personalization. In *NAACL*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. 2013. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *ICLR*.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2017. Modeling relational data with graph convolutional networks. In *ESWC*.

Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. 2019. End-to-end structure-aware convolutional networks for knowledge base completion. In *AAAI*.

Arkadii Slinko. 2000. *A generalization of Komlós theorem on random matrices*. Univ., Department of Mathematics, School of Mathematical and Information.

Zhiqing Sun, Zhi-Hong Deng, Jing Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *ICML*, pages 2071–2080.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph attention networks. In *ICLR*.

Eugene Vorontsov, Chiheb Trabelsi, Samuel Kadoury, and Christopher Joseph Pal. 2017. On orthogonality and learning recurrent networks with long term dependencies. In *ICML*.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *TKDE*, 29:2724–2743.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.

Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *KDD*.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. In *NeurIPS*.

A Discussion on the Ability of Pattern Modeling and Inference

It can be proved that *OTE* can infer all three types of relation patterns, e.g., symmetry/antisymmetry, inversion and composition patterns.

A.1 Symmetry/antisymmetry

If $e_t = f(r, h)$ and $e_h = f(r, t)$ hold, we have

$$\begin{aligned} e_t &= diag(\exp(s_r))\phi(M_r) \\ & \quad diag(\exp(s_r))\phi(M_r)e_t \\ \Rightarrow & \quad \phi(M_r)\phi(M_r) = I \\ & \quad s_r = 0 \end{aligned}$$

In other words, if $\phi(M_r)$ is a symmetry matrix and no scale is applied, the relation is symmetry relation.

If the relation is antisymmetry, e.g., $e_t = f(r, h)$ and $e_h \neq f(r, t)$, we just need to one of the $\phi(M_r(i))$ is not symmetry matrix or $s_r(i) \neq 0$.

A.2 Inversion

If $e_2 = f(r_1, e_1)$ and $e_1 = f(r_2, e_2)$ hold, we have

$$\begin{aligned} e_2 &= diag(\exp(s_{r_1}))\phi(M_{r_1}) \\ & \quad diag(\exp(s_{r_2}))\phi(M_{r_2})e_2 \end{aligned}$$

In other words, if $diag(\exp(s_{r_1}))\phi(Mr_1) = \phi(M_{r_2})^T diag(\exp(-s_{r_2}))$, the relation r_2 is inverse relation of r_1.

A.3 Composition

If $e_2 = f(r_1, e_1)$, $e_3 = f(r_2, e_2)$ and $e_3 = f(r_3, e_1)$ hold, we have

$$\begin{gathered} diag(\exp(s_{r_3}))\phi(M_3)e_1 = \\ diag(\exp(s_{r_2}))\phi(M_2) \\ diag(\exp(s_{r_1}))\phi(M_1)e_1 \end{gathered}$$

It means if $diag(\exp(s_{r_3}))\phi(M_3)$ is equal to $diag(\exp(s_{r_2}))\phi(M_2)diag(\exp(s_{r_1}))\phi(M_1)$ then relation r_3 is composition of relation r_1 and r_2.