Orthogonal Relation Transforms with Graph Context Modeling for Knowledge Graph Embedding

Yun Tang, Jing Huang, Guangtao Wang, Xiaodong He, and Bowen Zhou
JD AI Research

Abstract

Translational distance based knowledge graph embedding has shown progressive improvements on the link prediction task, from *TransE* to the latest state-of-the-art *RotatE*. However, N-1, 1-N and N-N predictions still remain challenging. In this work, we propose a novel translational distance based approach for knowledge graph link prediction. The proposed method includes two-folds, first we extend the *RotatE* from 2D complex domain to high dimension space with orthogonal transforms to model relations for better modeling capacity. Second, graph context is explicitly modeled via two directed context representations. These context representations are used as part of 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-1, 1-N and N-N cases for knowledge graph link prediction task. The experimental results show that it achieves better performance on two benchmark data sets compared to the baseline *RotatE*, especially on data set (FB15k-237) with many high in-degree connection nodes.

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, Biega, and Suchanek, 2013), have been built over the years and successfully applied to many domains such as search, recommendation and question answering. Knowledge graph has become an increasingly crucial component in machine intelligence systems. Although these knowledge graphs have already contained millions of entities and facts, they are far from complete compared to existing facts and newly added knowledge from the real world. Therefore knowledge graph completion is important topic and drawn great attentions from academic and industrial researchers.

Knowledge graph embedding represents entities and relations in continuous vector spaces. It is widely used for knowledge graph completion and could be roughly categorized into two classes (Wang et al., 2017): translational distance models and semantic matching models. Translational distance models measure the plausibility of a fact as the distance between two entities. It is usually done with relation dependent translational operations. 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), *TransR* (Lin et al., 2015) to the latest *RotatE* (Sun et al., 2019). Another thread of research focuses on similarity-based scoring functions and measures fact plausibility by matching latent semantics of entities and relations embodied in their vector space representations, for example, *DistMult* (Yang et al., 2014), *ConvE* (Dettmers et al., 2018), *ConvKB* (Nguyen et al., 2017), *CapsE* (Nguyen et al., 2019) and *QuatE* (Zhang et al., 2019).

Though great 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 issues. In Figure 1, relation "profession" demonstrates a N-to-N relation example and the corresponding edges are highlighted as green. Assuming triple (SergeiRachmaninoff, Profession, Pianist) is unknown and we want to evaluate the plausibility of this triple, the model needs to rank it against all triples by replacing "SergeiRachmaninoff" or "Pianist" with other entities in the knowledge graph. Entity "SergeiRachmaninoff" connected to multiple entities as head entity via relation "profession", while "Pianist" as a tail entity also can reach multiple entities through relation "profession". It makes the prediction hard because the mapping from certain entity-relation pair can lead to multiple different entities.

In this work, a novel translational knowledge graph embedding with graph context is proposed to alleviate the 1-to-N, N-to-1 and N-to-N issues. The proposed approach includes two parts. First we extend the *RotatE* modeling from 2D complex domain to high dimension space for better modeling capacity. Orthogonal transform embedding (*OTE*) employs orthogonal transforms to represent relations in knowledge graph. In addition, the entity embedding is divided into groups. Each group is modeled and scored independently. The final score is the summation of all group scores. Hence, each group could address different aspects of entity-relation pair and alleviate 1-to-N, N-to-1 and N-to-N issues. Second, graph context is used to integrate graph structure information in the knowledge graph. Considering the triple (SergeiRachmani-

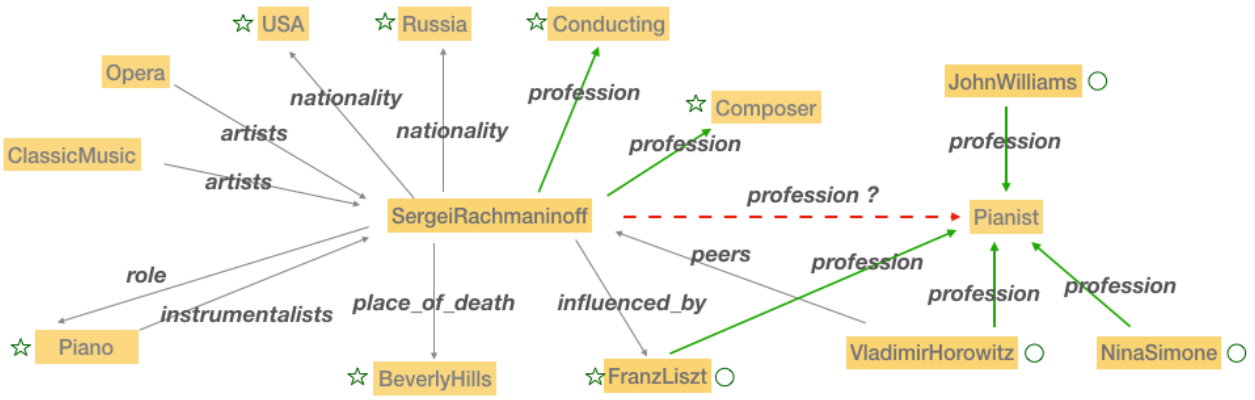

Figure 1: Snapshot of knowledge graph in FB15k-237. Entities are represented as golden blocks.

noff, Profession, Pianist): from the incomplete knowledge graph, human can 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* 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 translational model, orthogonal transform embedding *OTE*, is proposed to extend *RotatE* from 2D space to high dimensional space.
- A directed graph context modeling method is proposed to integrate knowledge graph structure, including both neighboring entity nodes and relation edges;
- Experimental results of *OTE* on standard benchmark FB15k-237 and WN18RR datasets show consistent improvements over *RotatE*, the state of art translational distance embedding model, especially on FB15k-237 with many high in-degree connection nodes.

Related work

Knowledge Graph Embedding

Translational distance model is also known as additive models, since it projects head and tail entities into the same embedding space and the difference between two entity embeddings is used to describe 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 the state-of-art within 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, Allen, and Hospedales, 2019) and *QuatE* (Zhang et al., 2019). *ConvKB* (Nguyen et al., 2017) and *CapsE* (Nguyen et al., 2019) are further integrating head, relation and tail embeddings from the beginning of modeling and the triple as a whole is measured together using convolutional model or capsule networks.

Those embedding methods focus on modeling the individual triple and achieve good performance for knowledge graph completion. However, they ignore knowledge graph structure and don't take the advantage of context from neighbouring nodes and connections. This issue inspired the usage of graph neural networks (Kipf and Welling, 2016; Veličković et al., 2017) for graph context modeling. Encoder-decoder framework is adopted in (Schlichtkrull et al., 2017; Shang et al., 2018; 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 models could be end-to-end based that the graph encoder and graph embedding decoder are training together. It could also be separated that the graph encoder output is only used to initialise the entity embedding in the graph embedding model (Nathani et al., 2019).

Orthogonal Transform

Orthogonal transform is considered to be more stable and efficient for neural networks (Saxe, McClelland, and Ganguli, 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, Chen, and Wang, 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).

Preliminaries

RotatE as an orthogonal transform

The proposed translational models used in this study are partially inspired by work *RotatE* (Sun et al., 2019). In *RotatE*, the embedding translation 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 image components, and the translational computation 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* shows good performance on graph embedding, it is defined in $2D$ complex domain and limited it's modeling ability. A natural extension is to apply similar operation on a higher dimension space and it is the motivation behind *OTE* described in section .

Gram Schmidt process

In this paper, we use Gram-Schmidt process to orthogonalize a linear transform into an orthogonal transform. The Gram-Schmidt process takes a set of tensor $S = \{v_1, ..., v_k\}$ for $k \leq d$ and generates an orthogonal set $S' = \{u_1, ..., u_k\}$ that spans the same $k-$dimensional subspace of $\mathcal{R}^d$ as S.

$$t_i = v_k - \sum_{j=1}^{k-1} \frac{< v_k, t_j >}{< t_j, t_j >} t_j \quad (1)$$

$$u_i = \frac{t_i}{||t_i||} \quad (2)$$

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

Orthogonal matrix has many desired properties for neural network based machine learning methods. For example, it is able to get the corresponding reverse matrix by transposing itself; it also preserves energy that the L_2 norm of vector is kept before and after transform. It is thought to be more stable and efficient in many neural networks based models (Bansal, Chen, and Wang, 2018). In this study, we are more interested in its property to obtain reverse matrix by transpose and it makes the reverse project operation discussed in section possible.

Methods

We consider knowledge graph as a directed graph $G = (V, R)$ throughout this section, where V is a set of entity nodes with $|V| = N$, and $R \subseteq V \times V$ is a set of relation edges with $|R| = M$. Facts are stored in a knowledge graph as a collection of triples $\mathcal{D} = \{(h, r, t)\}$. Each triple has a head entity h and tail entity $t \in V$. 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 relations (Bordes et al., 2013; Wang et al., 2014) are the main issues in the current systems. They are addressed in the proposed approach by: 1) *OTE* to handle the mapping from one entity-relation pair to different entities ; 2) Directed graph context to integrate knowledge graph structure information to reduce the uncertainty.

Orthogonal Transform Embedding (OTE)

Head, relation and tail entity are represented as e_h, M_r, e_t in *OTE*, where e_h, $e_t \in \mathcal{R}^d$, and d is the dimension of the entity embedding. The entity embedding e is further considered as concatenation of K sub-vectors, e.g., $e = [e(1); ...; e(K)]$, where $e(i) \in \mathcal{R}^{d_s}$ and $d = K\dot{d}_s$. M_r is a collection of K linear transform matrix $M_r = \{M_r(1), ..., M_r(K)\}$, and $M_r(i) \in \mathcal{R}^{d_s \times d_s}$. For each sub-embedding $e(i)$, we assume the projection from h and r to t is calculated as below:

$$\tilde{e}_t(i) = f_i(r, h) = \phi(M_r(i)) e_h(i) \quad (3)$$

where ϕ is Gram Schmidt process applied to square matrix $M_r(i)$ and 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 equation 3, e.g., $\tilde{e}_t = f(r, h) = [\tilde{e}_t(1); ...; \tilde{e}_t(K)]$. Equation 3 defines a simple transition between node embeddings via relation embedding. The L_2 norm of $e_h(i)$ is preserved before and after transform. Improving the training stability through orthogonal matrix is not priority in this study due to the shallowness of the model. Instead, it might limit the modeling ability. Hence, a scalar tensor $s_r(i) \in \mathcal{R}^{d_s}$ is introduced to match entity embeddings with different L_2 norms and Equation 3 is re-written as

$$f_i(r, t) = diag(\exp(s_r(i))) \phi(M_r(i)) e_h(i) \quad (4)$$

The corresponding distance scoring function is defined as

$$d(t|h, r) = \sum_{i=1}^{K} (||f_i(r, h) - e_t(i)||) \quad (5)$$

The reverse project from tail to head can be obtained by simply transposing the $\phi(M_r(i))$ and reversing the sign of s_r as below,

$$\bar{f}_i(r, t) = diag(\exp(-s_r(i))) \phi(M_r(i))^T e_t(i) \quad (6)$$

where $\bar{f}$ means the translational project function from tail entity and relation pair to the head entity.

The Gram Schmidt process is employed as part of computation graph in our model. $M_r(i)$ 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). One potential implementation issue is that $M_r(i)$ might not be an invertible matrix and it makes the compute $\phi(M_r(i))$ problematic. In our experiments, we initialize $M_r(i)$ to make sure they are with full rank. During training, we also keep checking the determinant of $M_r(i)$ and we found the update is fairly stable that we don't observe any issue for experiments with dimensions varied from 5 to 100 we explored.

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 .

Directed Graph Context

Head Relation Pair Context As discussed in the introduction section, the knowledge graph structure context could provide valuable information for link prediction task. In order to measure the plausibility of the questionable triple (h, r, t), the difference between $f(r, h)$ and e_t can be measured directly. In the other hand, assuming (h', r', t) is a valid triple in the training set, if the context pair (h', r') is similar to (h, r), it is a good indicator that the questionable triple (h, r, t) could be a valid one too. Hence, we can compare projects from other heads of t with project $f(r, h)$ from the questionable triple (h, r, t). All connected (head) entity-relation pairs (h', r') of t are considered as its graph context and denoted as $Ng(t)$. The questionable triple (h, r, t) gets a high score if we could find any of those valid projects from $Ng(t)$ are close to $f(r, h)$.

However, it is expensive to keep all context projects in $Ng(t)$ and compare them individually. In this work, an approximated approach is taken. For each tail entity node t, a head context representation $\tilde{e}_t^c$ is defined as the average from all corresponding context pairs in $Ng(t)$ as below:

$$\tilde{e}_t^c = \frac{\sum_{(h',r') \in Ng(t)} f(r', h') + e_t}{|Ng(t)| + 1} \tag{7}$$

The similarity between pair (h, r) from questionable triple (h, r, t) and head context of t is measured as

$$d_c(t|h, r) = \sum_{i=1}^{K} (||\tilde{e}_t(i) - \tilde{e}_t^c(i)||) \tag{8}$$

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

Tail Relation Pair Context Equation 5 only considers the difference between project of pair (h, r) to t, and it is natural also considered difference between project of pair (r, t) to h when we measure the plausibility of triple (h, r, t).

$$d(h|r, t) = \sum_{i=1}^{K} (||\tilde{e}_h(i) - e_h(i)||) \tag{9}$$

Similarly, we can compute the tail context representation $\tilde{e}_h^c$ for every head entity node h. The corresponding distance score between (r, t) pair from questionable triple (h, r, t) and tail context of h is defined as below:

$$d_c(h|r, t) = \sum_{i=1}^{K} (||\tilde{e}_h(i) - \tilde{e}_h^c(i)||) \tag{10}$$

We further combine all 4 distance scores discussed above as new distance score for the graph context orthogonal transform embedding (*GC-OTE*) training and inference

$$\begin{aligned} d_{all}(h, r, t) = {} & d(t|h, r) + d_c(h|r, t) \\ & + d(h|r, t) + d_c(t|h, r) \end{aligned} \tag{11}$$

Figure 1 demonstrates the computation of graph context for the questionable 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 graph context of "Pianist". The pairs in the head graph context are employed to calculate $\tilde{e}_h^c$ of "Pianist". While entities with ⋆ are tail entities for entity "SergeiRachmaninoff". Those entites and corresponding relations are the tail graph context of entity "SergeiRachmaninoff". Similarly, they are used to update $\tilde{e}_t^c$ of entity "SergeiRachmaninoff".

The generation of graph structure context defined in Equation 7 can be considered as a variant of one layer GCN (Kipf and Welling, 2016). Compared with previous proposed GNN based methods (Schlichtkrull et al., 2017; Shang et al., 2018; Bansal et al., 2019), our approach has 3 differences. First, the proposed method is based on directed graph instead of non-directed graph; second, in message passing phase, the proposed method employs entity-relation project in *OTE* to pass information from neighbouring nodes to the target node, while GCN or GAT has a separate matrix to transform the node embedding in message passing phase; third, the output of graph model is used for scoring function directly instead of as input for the following embedding method.

Experiments

Datasets

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

FB15k-237 The 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.

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.

WN18RR 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 dataset (Dettmers et al., 2018) is created to ensure that the evaluation dataset does not have test leakage due to redundant inverse relation. In summary, WN18RR dataset contains 93,003 triples with 40,943 entities and 11 relation types.

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.

Evaluation Protocol:

Following the evaluation protocol used in (Dettmers et al., 2018; Sun et al., 2019), each test triple (h, r, t) is measured into 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 and 10 (Hits@1, Hits@3, Hits@10), mean rank (MR) and the mean reciprocal rank (MRR) are reported in the experiments.

Experimental Setup

The *OTE* model hyper-parameters are determined by a grid search during the training, including learning rate, embedding size and sub-embedding dimension d_s. The best models are selected by early stopping on the validation set. The system is first trained with *OTE* or *RotatE* embedding and then the corresponding graph context based model is fine tuned on the pre-trained model.

For different datasets, we have found that the following settings work well: for FB15k-237, embedding size is set to 400, sub-embedding dimension to 20 and learning rates to $2e-3$ and $2e-4$ for pre-training and fine-tuning stages respectively ; for WN18RR dataset, embedding size is set to 400, sub-embedding dimension to 4 and learning rate to $1e-4$ and $3e-5$ for pre-training and fine-tuning stages. For *OTE* alone model, no fine-tuneing step is applied.

We use the adaptive moment (Adam) algorithm (Kingma and Ba, 2014) to train the model. Our models are implemented by PyTorch and run on NVIDIA Tesla P40 Graphics Processing Units. The graph neural networks implementation is based on Geometric PyTorch (Fey and Lenssen, 2019). 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 during training, the inference could be efficient if both head and tail context representations are precomputed and saved for each entity in the knowledge graph. [1]

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

$$\begin{aligned} L &= -\sum p(h', r, t') \log \sigma(d_{all}(h', r, t') - \gamma) \\ &\quad - \log \sigma(\gamma - d_{all}(h, r, t)) \end{aligned} \tag{12}$$

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

Experimental Results

Main Results In Table 2, the *GC-OTE* is compared with a number of strong baselines. For translational distance models, *TransE* (Bordes et al., 2013) and it's latest development *RotatE* (Sun et al., 2019) are compared; For semantic matching models, results from *DistMult* (Yang et al., 2014), *ComplEx* (Trouillon et al., 2016), *ConvE* (Dettmers et al., 2018), *TuckER* (Balazevic, Allen, and Hospedales, 2019) and *QuatE* (Zhang et al., 2019) are reported; Three systems with graph context information: *R-GCN+* (Schlichtkrull et al., 2017), *SACN* (Shang et al., 2018) and *A2N* (Bansal et al., 2019) are also included. All results except *OTE* and *GC-OTE* come from the corresponding literature.

On the FB15k-237 data set, the *GC-OTE* outperforms all models for all metrics except MR from *QuatE*. The MRR is improved from 0.338 in *RotatE*, to 0.361, or 7% improvement.

On the WN18RR data set, the *GC-OTE* also achieves slightly better MRR compared with the state of the art models *RotatE* and *QuatE*. 0.015 MRR improvement is observed from *RotatE* to *GC-OTE*.

In FB15k-237, there are average 18.7 relations per node, while this number drops to 2.1 edges per node in WN18RR. FB15k-237 has richer graph structure context compared with WN18RR. The results indicate that the proposed method *GC-OTE* is more effective at data set with rich structure context information.

Ablation Study In Table 3, an ablation study is conducted to examine the impact of each modification. The results from FB15k-237 validation set are reported. The embedding dimension for "*RotatE* S" and "*RotatE* L" are 400 and 2000 respectively[2]. "*RotatE* L" has the same configure as the model reported in (Sun et al., 2019). All other models are with embedding dimension 400. We examine the impact from model variation, sub-vector dimension (for *OTE*) and graph context.

First, when the embedding dimension increases from 400 to 2000, the MRR of *RotatE* is increased from 0.33 to 0.343

[1] We will release the source code after review.

[2] It is equivalent to the summation of dimensions from real and image part in *RotatE* setting.

Model	FB15k-237					WN18RR				
	MR	MRR	@1	@3	@10	MR	MRR	@1	@3	@10
TransE	357	.294	-	-	.465	3384	.226	-	-	.501
RotatE	177	.338	.241	.375	.533	3340	.476	.428	.492	.571
DistMult	254	.241	.155	.263	.419	5110	.43	.39	.44	.49
ComplEx	339	.247	.158	.275	.428	5261	.44	.41	.46	.51
ConvE	244	.325	.237	.356	.501	4187	.43	.40	.44	.52
QuatE	87	.348	.248	.382	.550	2314	.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	174	.351	.258	.388	.537	2968	.485	.437	.502	.587
GC-OTE	154	**.361**	**.267**	**.396**	**.550**	2715	**.491**	**.442**	**.511**	**.583**

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

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 test on FB15k-237 validation set.

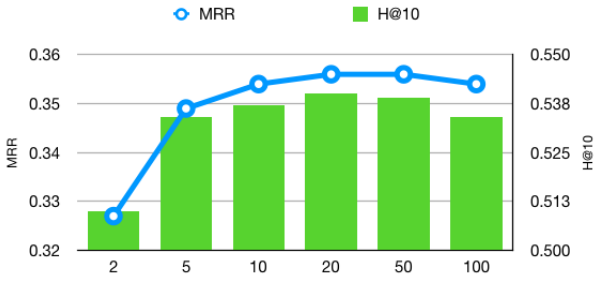

Figure 2: FB15k-237 for OTE with different sub-embedding dimension sizes.

(row 1 and 2). Second, for *OTE* models, when the sub-embedding dimension increases from 2 to 20, the MRR is improved from 0.327 to 0.355, though both models are with the same entity embedding dimension (row 3-4). *OTE* with sub-embedding dimension 20 outperforms both *RotatE* models listed in the table. Increasing the sub-embedding dimension helps to improve the modeling ability. It is more effective than increasing the embedding dimension size along (*RotatE* S v.s. L). Third, two variations of *OTE* are examined. Row 5 "*OTE* - scalar" is *OTE* model without diagonal scalar tensor as Equation 3. The MRR degrades slightly to 0.352. It indicates that preserving vector L_2 norm in the strict orthogonal transform limits the model ability in the shallow structure network. Row 6 "*LNE*" is a model using normal linear transform instead of orthogonal version. Hence, two types of relation transforms are estimated for projects from head to tail and tail to head respectively. The result is slightly worse than *OTE* and with significant more parameters. Last, row 7 and 8 show that graph context improve the performance further for the translational distance models without adding extra parameters. MRRs are increased by 0.014 and 0.011 for *RotatE* and *OTE* respectively.

In Figure 2, the impact of sub-embedding dimension to the *OTE* performance is demonstrated. The blue line shows MRR value for each sub-embedding dimension and green bars are the corresponding $H@10$ value. Both values increase and slowly saturated around 20. When the size gets bigger, the performance get worse though more parameters are used. The similar experiments are also conducted on WN18RR data set and we find the best sub-embedding dimension is 4 on that data set.

Error Analysis We split the triples in the FB15k-237 evaluation set into different categories to study the errors on 1-N, N-1 and N-N relations. The results (in H@10) are presented in Table 4. 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 defined below:

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

In Table 4, results from two models, *RotatE L* and *GC-OTE*, are compared for each category. "num." is the number of triples in the validation set belonging to the corresponding category; "H" or "T" is the experiment to predict head entity or tail entity given other entity and relation. "A" is the average result from both "H" and "T" experiments.

It is clear from the table that first prediction to entity in "N" side is harder than prediction to entity in the "1"

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

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

side. It is within our expectation. For example, given triple (SergeiRachmaninoff, Gender, male), it is easy to get high rank for tail entities "male" and "female", since they are quite different from other entities in the knowledge graph; but it is difficult to get a high rank for entity "SergeiRachmaninoff", since it has to compete with many other person names in the data set. Entity in the "N" side is easier to get confused with others and get a low rank. Second, *GC-OTE* improves for both "H" and "T" experiments at all categories compared with "*RotatE* L". It is also clear that the proposed approach has more gain on hard cases, say prediction at entities on the "N" side. It shows the graph context and orthogonal transform embedding do help on those hard cases.

Conclusions

In this work, a novel translational distance based approach for knowledge graph link prediction is proposed. It includes two-folds.

First, an orthogonal transform relation based graph embedding method *OTE* is proposed. *OTE* extends the modeling in *RotatE* from 2D complex domain to high dimension space with orthogonal relation matrix generated from Gram Schmidt process. The results show increasing the modeling dimension is more effective than increasing the embedding dimensionality.

Second, graph structure context is explicitly modeled via two directed context representations. Each node embedding in knowledge graph is augmented with two context representations, computed from the neighboring outgoing and incoming nodes respectively. These context representations are used as part of 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-1, 1-N and N-N issues within the knowledge graph link prediction tasks. The experimental results show that it achieves good performance in two common data sets compared with the baseline *RotatE*, especially on data set (FB15k-237) with rich structure information.

References

Auer, S.; Bizer, C.; Kobilarov, G.; Lehmann, J.; Cyganiak, R.; and Ives, Z. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*. Springer. 722–735.

Balazevic, I.; Allen, C.; and Hospedales, T. 2019. TuckER: Tensor factorization for knowledge graph completion. In *EMNLP*.

Bansal, T.; Juan, D.-C.; Ravi, S.; and McCallum, A. 2019. A2N: Attending to neighbors for knowledge graph inference. In *ACL*.

Bansal, N.; Chen, X.; and Wang, Z. 2018. Can we gain more from orthogonality regularizations in training deep networks? In *NeurIPS*.

Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; and Taylor, J. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 1247–1250. ACM.

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. In *NIPS*, 2787–2795.

Carlson, A.; Betteridge, J.; Kisiel, B.; Settles, B.; Hruschka Jr, E. R.; and Mitchell, T. M. 2010. Toward an architecture for never-ending language learning. In *AAAI*, volume 5, 3. Atlanta.

Dettmers, T.; Minervini, P.; Stenetorp, P.; and Riedel, S. 2018. Convolutional 2D knowledge graph embeddings. In *AAAI*.

Fey, M., and Lenssen, J. E. 2019. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Harandi, M., and Fernando, B. 2016. Generalized backpropagation, étude de cas: Orthogonality. *ArXiv*.

Huang, L.; Liu, X.; Lang, B.; Yu, A. W.; Wang, Y.; and Li, B. Q. 2017. Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks. In *AAAI*.

Ji, G.; He, S.; Xu, L.; Liu, K.; and Zhao, J. 2015. Knowledge graph embedding via dynamic mapping matrix. In *ACL*.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. In *ICLR*.

Kipf, T. N., and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; and Zhu, X. 2015. Learning entity and relation embedings for knowledge graph completion. In *AAAI*, volume 15, 2181–2187.

Mahdisoltani, F.; Biega, J.; and Suchanek, F. M. 2013. Yago3: A knowledge base from multilingual wikipedias. In *CIDR*.

Nathani, D.; Chauhan, J.; Sharma, C.; and Kaul, M. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs. In *ACL*.

Nguyen, D. Q.; Nguyen, T. D.; Nguyen, D. Q.; and Phung, D. 2017. A novel embedding model for knowledge base completion based on convolutional neural network. *arXiv preprint arXiv:1712.02121*.

Nguyen, D. Q.; Vu, T.; Nguyen, T. D.; Nguyen, D. Q.; and Phung, D. 2019. A Capsule Network-based Embedding Model for Knowledge Graph Completion and Search Personalization. In *NAACL*, 2180–2189.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in pytorch.

Saxe, A. M.; McClelland, J. L.; and Ganguli, S. 2013. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *ICLR*.

Schlichtkrull, M. S.; Kipf, T. N.; Bloem, P.; van den Berg, R.; Titov, I.; and Welling, M. 2017. Modeling relational data with graph convolutional networks. In *ESWC*.

Shang, C.; Tang, Y.; Huang, J.; Bi, J.; He, X.; and Zhou, B. 2018. End-to-end structure-aware convolutional networks for knowledge base completion. In *AAAI*.

Sun, Z.; Deng, Z.-H.; Nie, J.; and Tang, J. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR*.

Toutanova, K., and Chen, D. 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*, 57–66.

Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; and Bouchard, G. 2016. Complex embeddings for simple link prediction. In *ICML*, 2071–2080.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2017. Graph attention networks. In *ICLR*.

Vorontsov, E.; Trabelsi, C.; Kadoury, S.; and Pal, C. J. 2017. On orthogonality and learning recurrent networks with long term dependencies. In *ICML*.

Wang, Z.; Zhang, J.; Feng, J.; and Chen, Z. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, volume 14, 1112–1119.

Wang, Q.; Mao, Z.; Wang, B.; and Guo, L. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29:2724–2743.

Yang, B.; Yih, W.-t.; He, X.; Gao, J.; and Deng, L. 2014. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.

Zhang, S.; Tay, Y.; Yao, L.; and Liu, Q. 2019. Quaternion knowledge graph embedding. *CoRR* abs/1904.10281.

Disussion 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.

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 $s_r(i) \neq 0$.

Reversion:

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.

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

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

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.