A Vectorized Relational Graph Convolutional Network for Multi-Relational Network Alignment

Rui Ye[1] , **Xin Li**[*1] , **Yujie Fang**[1] , **Hongyu Zang**[1] and **Mingzhong Wang**[2]
[1]School of Computer Science, Beijing Institute of Technology, China
[2]USC Business School, University of the Sunshine Coast, Australia
{yerui, xinli, fangyujie, zanghyu}@bit.edu.cn, mwang@usc.edu.au

Abstract

Alignment of multiple multi-relational networks, such as knowledge graphs, is vital for AI applications. Different from the conventional alignment models, we apply the graph convolutional network (GCN) to achieve more robust network embedding for the alignment task. In comparison with existing GCNs which cannot fully utilize multi-relation information, we propose a vectorized relational graph convolutional network (VR-GCN) to learn the embeddings of both graph entities and relations simultaneously for multi-relational networks. The role discrimination and translation property of knowledge graphs are adopted in the convolutional process. Thereafter, AVR-GCN, the alignment framework based on VR-GCN, is developed for multi-relational network alignment tasks. Anchors are used to supervise the objective function which aims at minimizing the distances between anchors, and to generate new cross-network triplets to build a bridge between different knowledge graphs at the level of triplet to improve the performance of alignment. Experiments on real-world datasets show that the proposed solutions outperform the state-of-the-art methods in terms of network embedding, entity alignment, and relation alignment.

1 Introduction

Network alignment has recently attracted special attention from both industry and academia. It aims at finding the corresponding network elements (referred to as anchors thereafter), such as nodes or relations, in different networks if they refer to the same entity/relationship. The alignment of networks, such as aligning multi-lingual knowledge graphs, helps to construct more complete and compact networks, thus increasing the accuracy and robustness for applications like knowledge inference and cross-domain recommendation.

Embedding-based models are promising and efficient solutions for network alignment tasks because they are capable of representing and preserving the structures of networks in low-dimensional subspaces. Thereafter, the alignment can be achieved in the embedding spaces by sharing or transforming the embeddings of anchors.

Recent research in embedding models focuses on developing representation learning algorithms to acquire numerical representations of networks. Single-relational networks, in which all edges are of the same type, require to embed only nodes by representing each node i as a vector $u_i \in R^d$. In comparison, multi-relational networks, in which the vertices are connected by various types of edges or relations, require to embed both nodes and relations as low-dimensional vectors to preserve the network structure.

In general, two categories of embedding models can be summarized based on whether considering the embedding of relations. For example, DeepWalk [Perozzi *et al.*, 2014], and Node2Vec [Grover and Leskovec, 2016] target at single-relational networks and belong to "*embedding without relations*". In comparison, TransE [Bordes *et al.*, 2013] and its extensions [Wang *et al.*, 2014] model relations as translating operations on the low-dimensional embedding of the entities. That is, if there is an edge (h, r, t), in which a relation r links the head entity h and the tail entity t, then the equation $h + r = t$ should be preserved in the embedded space. They belong to "*embedding with relations*".

Motivated by the success of deep learning, increasing interests have emerged to apply task-driven deep network embedding for graph data [Bruna *et al.*, 2013; Defferrard *et al.*, 2016]. [Kipf and Welling, 2016] utilized a two-layer graph convolutional network (GCN) to achieve semi-supervised classification of graph nodes. With the integration of random walk or attention mechanism, [Niepert *et al.*, 2016] and [Ying *et al.*, 2018] proposed more dedicated GCN models for single-relational network embedding. However, the conventional spectral-based GCN and its variants can only process undirected single-relational networks because they require normalized graph Laplacian to be real symmetric positive semidefinite to facilitate graph Fourier transform, indicating that the adjacency matrix must be symmetric, and the 2-D adjacency matrix also restricts edges to be the same type.

To deal with multiple relation types, R-GCN [Schlichtkrull *et al.*, 2017] utilizes more representative node embeddings with the consideration of the effects of different relation types by iteratively accumulating weighted neighbor entities. How-

[*]Xin Li is the corresponding author. This work has been partially supported by National Key R&D Program of China under Grant No. 2017YFB0803300, NSFC under Grant No. 61772074.

ever, it has no relation representations involved.

In summary, existing GCN-based models belong to "*embedding without relations*". It is evident that relations in networks also carry rich semantic information as nodes, and the incorporation of relation embeddings would help to boost the effectiveness of relevant applications, such as knowledge base completion and multi-lingual knowledge graph alignment. Therefore, conventional GCNs and R-GCN cannot be applied for multi-relational network alignment tasks.

Addressing this issue, in this paper, we propose a vectorized relational graph convolutional network (VR-GCN), which generates both entity embeddings and relation embeddings simultaneously, to enable the incorporation of GCN and multi-relational networks. The design of its convolution function should consider the following three criterions:

1. Explicit relation embedding: VR-GCN should explicitly include the vectorized relation, and the relation embeddings will affect the learning of the entity embeddings.
2. Role discrimination: VR-GCN should be capable of performing different convolution operations on an entity according to its role. Generally, the same entity may take the role of being a head role in one triplet but of a tail in another. This requirement becomes indispensable for directed graphs.
3. Translation adoption: VR-GCN should conform to the translation operation $h+r \approx t$ to imply the network edge which has two entities h and t connected by a specific relation r.

Based on the network embedding, graph alignment can be achieved in the subspaces. Graph alignment, in fact, consists of entity alignment and relation alignment. However, most existing models for multi-relational network alignment, no matter they include relation embedding or not, either ignore the needs of relation alignment or assume all relations are previously aligned. Therefore, we propose an alignment framework, with VR-GCN as the network embedding model, to support both entity alignment and relation alignment. Experiments demonstrate that the inclusion of relation alignment, in turn, improves the outcomes of entity alignment.

The main contributions of this paper include:

1. We proposed VR-GCN for the embedding of multi-relational networks, in which both entity embeddings and relation embeddings are explicitly supported. VR-GCN combines the strength of both convolutional and translational properties for multi-relational networks.
2. We proposed AVR-GCN, the alignment framework based on VR-GCN, to support both entity alignment and relation alignment for multi-relational networks. Optimization techniques are designed to improve the effectiveness of the alignment results.
3. We evaluated the proposed alignment framework with entity alignment, relation alignment, and link prediction tasks on knowledge graphs. The results demonstrate that our method constantly outperforms other state-of-the-art approaches for various tasks, and prove that the inclusion of relation alignment can improve the outcomes of entity alignment.

2 Related Work

Our work is mainly related to two lines of research: network embedding and knowledge graph alignment.

2.1 Network Embedding

Network embedding refers to learning representations of nodes/edges in networks, which has demonstrated its effectiveness in many application scenarios. For single-relational networks, DeepWalk and Node2vec extend the skip-gram model in natural language processing to the representation learning of networks. LINE attempts to generate the representations of nodes by retaining the "first-order proximity" and/or "second-order proximity" of local structures. For multi-relational networks, TransX [Bordes *et al.*, 2013; Wang *et al.*, 2014; Lin *et al.*, 2015] interprets relations as translating operations between head and tail nodes, which can be denoted as $h + r \approx t$.

Motivated by the powerful feature extraction capability and remarkable success of deep neural networks, [Defferrard *et al.*, 2016] proposed a spectral graph theoretical formulation of CNNs on graphs and a convolutional network extending the conventional CNNs to non-Euclidean space. [Kipf and Welling, 2016] further extended this idea and proposed graph convolutional neural networks (GCNs) to integrate the connectivity patterns and feature attributes of graph-structured data, and achieved remarkable results in semi-supervised classification. Thereafter, a series of improvements and extensions were proposed based on GCN. GAT [Velickovic *et al.*, 2017] employs the attention mechanism to GCNs, in which each node gets an importance score based on its neighborhood, thus providing more expressive representations for nodes. R-GCN is a relation-aware node embedding model based on GCN for knowledge graphs. The convolution of R-GCN is a relation-specific transformation in which the impact weights between different nodes depend on the relation types.

Although R-GCN applies the relation information as the weights to induce node embedding, it does not support the representations of relations themselves. In fact, the vector representations of relations are critical in tasks like relation alignment of Knowledge graphs. Moreover, the proper representation learning of relations may help to improve the quality of node representations.

2.2 Knowledge Graph Alignment

The alignment of multi-relational networks centers on aligning knowledge graphs. It is generally accomplished by using network embeddings to identify the linkage between networks via the structural and anchor information.

MTransE [Chen *et al.*, 2017] embeds entities and relations of each knowledge graph in a separate space with TransE, and then provides five different variants of transformation functions to project the embedded vectors from one subspace to another. The candidate set of one node's correspondence in the other network can be obtained by ranking the distance between them in the transformed space. In contrast, ITransE [Zhu *et al.*, 2017] utilizes TransE to learn one common low-dimensional subspace for all knowledge graphs, with the constraint that the observed anchor seeds from dif-

ferent knowledge graphs share the same vector representation in the subspace. Furthermore, ITransE iteratively updates the embeddings of entities and relations to perform the alignment when the "confidence" of a newly discovered anchor pair is larger than an empirical threshold. AlignE [Sun *et al.*, 2018] also adopts TransE to learn network embeddings, and applies parameter swapping to encode networks into a unified space. Instead of using TransX to direct the representation learning of knowledge graph, NTAM [Li *et al.*, 2018] employs a probabilistic embedding model [Liu *et al.*, 2017] to learn the node embeddings for alignment tasks. It claims to retain the triangular structures in networks that TransX fails. GCN(SE) [Wang *et al.*, 2018] levers on the conventional GCN to encode the entities from multiple knowledge graphs into a unified embedding space to perform the alignment. It views the relations as influential weights on entities in proportion to the number of entities connected with different types of relations.

However, the aforementioned alignment approaches focus on entity alignment across networks without explicit relation representations [Wang *et al.*, 2018] or assuming the relations are previously fully aligned [Zhu *et al.*, 2017]. Therefore, they are inherently insufficient or incapable of relation alignment tasks, which we argue is also an important task as illustrated in Sec. 1. In comparison, we propose a vectorized relational GCN (VR-GCN) to learn the embeddings of both nodes and relations explicitly for knowledge graphs. With both entity and relation representations learned, the alignment framework could be optimized by minimizing the "entity loss" as well as "relation loss", which is semi-supervised by the observed entity anchor pairs and relation anchor pairs.

3 Model

A Knowledge Graph $KG = (E, R, T)$ is a directed network, where E, R and T denote the set of entities, relations and triplets respectively. A triplet (h, r, t) represents a tail entity t is connected to the head entity h by the relation r. Knowledge Graph Alignment finds the aligned entities and relations between $KG_i = (E_i, R_i, T_i)$ and $KG_j = (E_j, R_j, T_j)$. The prior knowledge of the aligned entities/relations are observed anchors. They are denoted as the set of anchor entity pairs $E_a = \{(e_i, e_j)|e_i \in E_i, e_j \in E_j\}$ and the set of anchor relation pairs $R_a = \{(r_i, r_j)|r_i \in R_i, r_j \in R_j\}$ respectively.

3.1 VR-GCN Framework

In comparison with existing GCNs, VR-GCN is capable of learning the vectorized embedding of relations, in addition to that of entities, in the convolution process.

VR-GCN integrates the strength of GCN and the translational property in knowledge graphs [Bordes *et al.*, 2013] $(h + r \approx t)$ to design the new propagation model. Eq.(1) defines the update rule for the entity and relation embeddings:

$$h_i^{l+1} = \sigma((\sum_{r\in N_r}\sum_{t\in N_t^r} c(h_t^l, h_r^l) + \sum_{r\in N_r}\sum_{h\in N_h^r} \hat{c}(h_h^l, h_r^l) + h_i^l)W^l) \quad (1)$$

where N_r denotes the set of relations connecting the entity i. N_t^r denotes the set of tail entities connected with the entity i by the relation r. N_h^r denotes the set of head entities

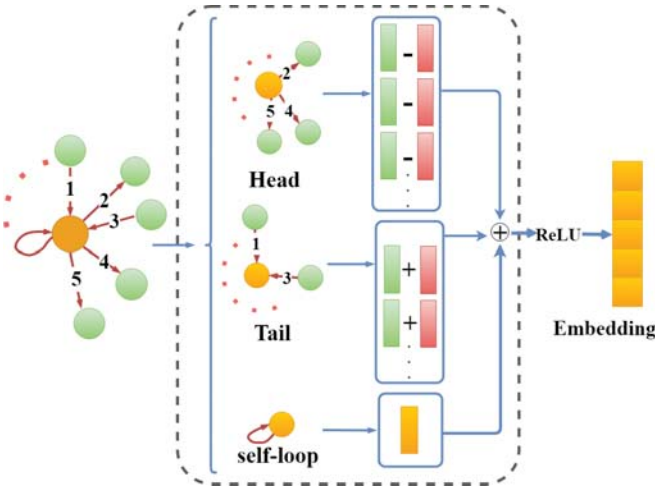


Figure 1: Embedding process in VR-GCN. Yellow represents the center entity, red represents relations connected with it, and green represents its neighboring entities. If the entity has the head role, accumulating its neighboring tail nodes and relations with $t - r$; If it has the tail role, accumulating its neighboring head nodes and relations with $h + r$. The role discrimination representations are accumulated in a (normalized) sum and passed through a ReLU function. Meanwhile, the embeddings of relations are also updated.

connected with the entity i by the relation r. $h_h^l \in R^{d(l)}$, $h_r^l \in R^{d(l)}$, and $h_t^l \in R^{d(l)}$ denote the l^{th} layer embedding of the head entity, relation, and tail entity respectively in the neural network, and $d(l)$ is the dimensionality of this layer's representation. σ is the activation function, such as Sigmoid and Relu. $c(\cdot,\cdot)$ is the function to describe the relationship between h_t^l and h_r^l, and $\hat{c}(\cdot,\cdot)$ describes the relationship between h_h^l and h_r^l. W^l is the weight matrix of the l^{th} layer.

Eq.(1) features the role discrimination (Section 1) criterion to identify if entity i in the knowledge graph takes the role of head or tail entity regarding a specific relation r. It performs different convolution operations for them: if i has the head entity role, its embedding is calculated by combining the related tail entity h_t^l and relation h_r^l; Otherwise, its embedding is calculated with the related head entity h_h^l and relation h_r^l. Thereafter, all occurrences of head roles and tail roles of i are added, together with a single self-connection representation h_i^l, to infer the $l + 1$ representation of the entity i.

The design of function c and $\hat{c}$ features the translation adoption criterion which is $h + r \approx t$ for a triplet (h, r, t) in the graph. Alternatively, the translational property can be transformed into $h \approx t - r$ and $t \approx h + r$. Therefore, let

$$c(h_t^l, h_r^l) = (h_t^l - h_r^l) \quad (2a)$$

$$\hat{c}(h_h^l, h_r^l) = (h_h^l + h_r^l) \quad (2b)$$

Apply them to Eq.(1), the convolutional function becomes:

$$h_i^{l+1} = \sigma((\frac{1}{d_i}(\sum_{r\in N_r}\sum_{t\in N_t^r} (h_t^l - h_r^l) + \sum_{r\in N_r}\sum_{h\in N_h^r} (h_h^l + h_r^l)) + h_i^l)W^l) \quad (3)$$

where d_i is the normalization coefficient which equals to the degree of the entity i, including its outdegree and indegree.

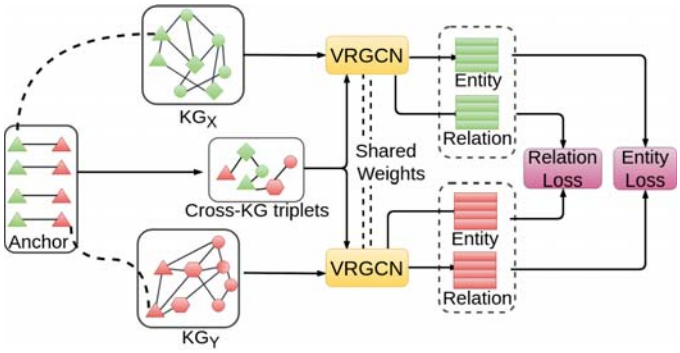


Figure 2: AVR-GCN: Knowledge Graph Alignment Framework based on VR-GCN

Fig.1 illustrates the embedding process defined by Eq.(3). It updates the embedding of an entity by accumulating the embeddings of its neighboring entities and relations, and updates the embedding of a relation by integrating embeddings of its head and tail entities. It can be applied to a neural network to update embeddings of entities and relations, and multiple layers can acquire more flexible structure information.

The proposed convolutional function is capable of distinguishing different roles of entities and utilizing the translational property in knowledge graphs to learn the embeddings of both entities and relations, which in fact helps to retain the structural information of the graphs. In comparison with existing GCNs, the vectorized representations of relations have better support for knowledge graph alignment tasks.

3.2 VR-GCN-based Knowledge Graph Alignment

This section introduces AVR-GCN, a knowledge graph alignment framework based on VR-GCN.

As illustrated in Fig.2, given two knowledge graphs KG_X and KG_Y, each of them learns its embedding representation based on VR-GCN first to capture the structural information of each graph. Denote their embeddings as VR-GCN$_X$ and VR-GCN$_Y$ respectively. Since each graph has a unique embedding space, the weight sharing mechanism [Wang *et al.*, 2018] is utilized to join them into a unified space for alignment. By sharing the weight matrices $W^{(X)}$ and $W^{(Y)}$, the problem of estimating the probability of entities/relations alignment is converted into measuring the distance between entities/relations in the unified embedding space.

Anchors are the knowledge about the aligned entity or relation pairs from different graphs. In this framework, anchor information is applied in two ways to enhance the alignment performance. It is first utilized to define the objective function for the alignment. In the shared embedding space, the distances between aligned entities/relations should be minimized and those of unaligned entities/relations should be maximized. Therefore, the objective function is:

$$O = \sum_{(e_x,e_y)\in E_a} \sum_{(e'_x,e'_y)\notin E_a} [d(\boldsymbol{e_x},\boldsymbol{e_y}) + \xi - d(\boldsymbol{e'_x},\boldsymbol{e'_y})]+ \\ \alpha \sum_{(r_x,r_y)\in R_a} \sum_{(r'_x,r'_y)\notin R_a} [d(\boldsymbol{r_x},\boldsymbol{r_y}) + \xi - d(\boldsymbol{r'_x},\boldsymbol{r'_y})] \quad (4)$$

Dataset	DBP$_{ZH\text{-}EN}$	DBP$_{JA\text{-}EN}$	DBP$_{FR\text{-}EN}$
#aligned entity	15,000	15,000	15,000
#aligned relation	891	592	814
#Entity	19,388(ZH) 19,572(EN)	19,814(JA) 19,780(EN)	19,661(FR) 19,993(EN)
#Relation	1,701(ZH) 1,323(EN)	1,298(JA) 2,451(EN)	1174(FR) 1,208(EN)
#Triplet	70,414(ZH) 95,142(EN)	77,214(JA) 93,484(EN)	105,998(FR) 115,772(EN)

Table 1: Dataset summary

where (e_x, e_y) and (r_x, r_y) denote a pair of aligned entities and relations respectively, and (e'_x, e'_y) and (r'_x, r'_y) denote its corresponding unaligned (negative) pair of entities and relations. Boldfaced $\boldsymbol{e_x}, \boldsymbol{e_y}, \boldsymbol{r_x}, \boldsymbol{r_y}, \boldsymbol{e'_x}, \boldsymbol{e'_y}, \boldsymbol{r'_x}, \boldsymbol{r'_y}$ represent the corresponding vectorized embeddings acquired by VR-GCN for aforementioned graph elements. $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_1$. ξ is a margin hyper-parameter separating positive and negative alignments, and α is also a hyper-parameter balancing the importance of the entity and relation alignment.

The second way is to use existing entity anchors to generate new cross-network triplets to build a bridge between different knowledge graphs at the level of triplet. Let T_x and T_y be the triplet sets for knowledge graph KG_X and KG_Y respectively. Following [Sun *et al.*, 2018], we generate the supervised triplets as:

$$T_a = \{(e_y, r, t)|(e_x, r, t) \in T_x\} \cup \{(h, r, e_y)|(h, r, e_x) \in T_x\} \\ \cup \{(e_x, r, t)|(e_y, r, t) \in T_y\} \cup \{(h, r, e_x)|(h, r, e_y) \in T_y\} \quad (5)$$

After injecting the generated triplet set T_a into both graphs, KG_X and KG_Y are expanded to have more shared edges. Consequently, their embeddings would be closer to each other, which will help to improve the alignment performance.

Putting them together, the following strategy is adopted:

(1) Randomly select a subset of anchors to generate T_a. Then, inject T_a to graphs before learning their embeddings with VR-GCN.

(2) With the learned embeddings, use the remaining anchors to minimize Eq.(4).

4 Experiment

We first evaluated AVR-GCN framework on cross-lingual entity alignment and relation alignment tasks to testify its effectiveness on multi-relational network alignment. In addition, VR-GCN model was experimented with link prediction task for its effectiveness on multi-relational network embedding.

4.1 Knowledge graph alignment

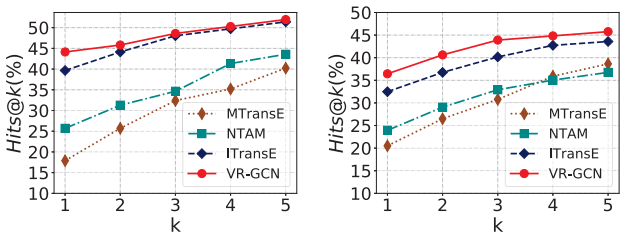
The experiments consist of cross-lingual entity alignment and relation alignment on the trilingual datasets, which were extracted from the real knowledge graph DBpedia containing multilingual versions of DBP$_{ZH-EN}$ (Chinese to English), DBP$_{JA-EN}$ (Japanese to English) and DBP$_{FR-EN}$ (French to English). Table 1 lists their statistical summaries.

Dataset	ZH-EN&EN-ZH				JA-EN&EN-JA				FR-EN&EN-FR			
Metrics(%)	$Hits@1$	$Hits@5$	$Hits@10$	MRR	$Hits@1$	$Hits@5$	$Hits@10$	MRR	$Hits@1$	$Hits@5$	$Hits@10$	MRR
MTransE	13.46	31.44	41.45	23.22	13.02	29.45	38.80	21.88	7.00	21.76	31.81	14.64
ITransE	21.94	45.90	54.77	32.88	17.02	39.95	48.74	27.57	12.46	34.10	43.51	22.50
NTAM	25.01	53.45	62.55	33.02	22.10	50.00	58.28	28.23	20.43	51.10	62.30	27.34
AlignE	31.78	59.21	69.43	45.25	30.34	58.25	69.88	43.30	32.60	63.54	74.92	46.65
GCN(SE)	26.00	54.88	64.69	38.96	27.05	56.47	66.10	40.03	27.25	56.84	67.96	40.58
GCN(SE)*	31.02	60.12	69.37	43.87	32.03	59.83	69.63	46.78	32.20	61.79	72.45	45.51
AVR-GCN(rl.exl.)	33.20	61.54	70.50	48.26	32.34	60.33	69.36	44.81	33.43	64.96	74.51	47.45
AVR-GCN	**37.96**	**64.60**	**73.27**	**50.19**	**35.15**	**61.66**	**72.15**	**47.03**	**36.06**	**66.11**	**75.14**	**49.46**

Table 2: Performance comparison on entity alignment

Metrics \ Ratio	0	0.2	0.4	0.6	0.8	1.0
$Hit@1(\%)$	34.30	34.41	35.16	37.72	38.77	39.34
$Hit@5(\%)$	61.97	62.39	62.87	64.89	65.53	66.13
$Hit@10(\%)$	70.25	70.61	71.55	73.13	73.87	75.06

Table 3: Training ratio of aligned relations vs. entity alignment



(a) ZH-EN (b) JA-EN

Figure 3: Performance comparison for relation alignment

Evaluation Metrics. Standard metrics, $Hits@k$ and mean reciprocal rank (MMR), were used to evaluate the performance of knowledge graph alignment. $Hits@k$ measures the proportion of correct items in top-k ranked candidates and MMR measures the mean reciprocal rank of correct entities and relations. Regarding the bi-directional feature of network alignment, which means the alignment may start with either network as the source, $Hits@k$ in this paper is computed as the average of iterating each network as the source.

Baselines and Settings. For the model configuration, the input feature vectors of entities and relations were randomly initialized, and then fed to the 2-layer VR-GCN to update the embeddings. In the experiments, $d = 300$, $\xi = 3$, $\alpha = 1$,and the ratio of anchors for new triplet generation was 0.5.

The following state-of-the-art approaches were included in the comparison for the graph alignment task:

- **MTransE** uses linear transformations between two vector spaces which are built by TransE.
- **ITransE** embeds entities from different graphs into a unified vector space and uses the predicted anchors iteratively to improve performance.
- **NTAM** utilizes a probabilistic model to obtain network embeddings to accomplish the alignment task.
- **AlignE** takes ε-truncated uniform negative sampling and parameter swapping to implement KG embedding. It is a variant of BootEA without bootstrapping.
- **GCN(SE)** uses structural information to align entities based on the entity embeddings learned by GCNs.
- **GCN(SE)*** is an extension to GCN(SE) by running GCN(SE) on the alignment framework proposed in this paper. We provided GCN(SE)* for a fair comparison.
- **AVR-GCN(rl.exl.)** is a variant of AVR-GCN but with the relation component in the objective function excluded. It serves as a baseline to differentiate the impact of relation alignment on entity alignment.

Entity Alignment

Each dataset was split into training and test sets as 30-70 which used less prior anchor information for the learning process. Moreover, for a given entity, the lookup scope for aligning candidates should include the entire network. Otherwise, if only known anchor entities are included in the lookup process, which was used in some existing models, the results may not be practical because of its implicit assumption that all anchors are known and accurate.

Table 2 reports the results, showing that AVR-GCN outperforms all other methods on all trilingual datasets. Specifically,

- AVR-GCN is significantly better than most translation-based methods, such as MTransE and ITransE. It proves that VR-GCN has better support for capturing the complex structure information in multi-relational networks than translation-based models. Besides, NTAM, which utilizes a probabilistic embedding space, also performs better than MTransE and ITransE, supporting the argument that TransX is inefficient in capturing complex network information.
- GCN(SE)* performs better than GCN(SE), showing that the proposed alignment framework is beneficial for graph alignment tasks.
- AVR-GCN(rl.exl.) outperforms other baselines even though no relation anchor information is applied in the alignment stage, indicating that the explicit consideration of relation embedding helps to improve entity alignment. Meanwhile, AVR-GCN is better than AVR-GCN(rl.exl.). The only difference between them is the

dataset	WN18					FB15k-237				
metrics	MRR		Hits@			MRR		Hits@		
	Raw	Filter	1	3	10	Raw	Filter	1	3	10
TransE	0.335	0.454	0.064	0.803	0.916	0.143	0.210	0.146	0.222	0.330
DisMult	0.540	0.829	0.726	0.923	0.940	0.127	0.220	0.144	0.238	0.369
R-GCN	0.526	0.773	0.650	0.889	0.944	0.138	0.225	0.133	0.249	0.423
VR-GCN	**0.565**	**0.847**	**0.764**	**0.929**	**0.946**	**0.155**	**0.248**	**0.159**	**0.272**	**0.432**

Table 4: Link prediction performance comparison in individual networks

inclusion of relation anchors in the alignment stage. Therefore, it is evident that the aligned relation information can boost the alignment performance.

- Finally, in comparison with GCN(SE)* which likewise utilizes GCN and the same alignment framework, VR-GCN boosts the performance by up to 22%, proving that the vectorized relation representations or relation embeddings are valuable for alignment tasks.

To evaluate the correlation between entity alignment and relation alignment, we incrementally increased the ratio of known aligned relations in the training process on the one-side ZH-EN dataset. It actually demonstrates the changes when we evolve from AVR-GCN(rl.exl.), which includes no relation alignment, to AVR-GCN, which has all relation alignment. Table 3 shows that the more previously known aligned relations included, the better performance for entity alignment. The observation validates the importance of relation alignment which has not attracted enough attention. In other words, AVR-GCN is capable of levering on the relation alignment to not only support relation alignment tasks, but also improve the performance of entity alignment.

Relation Alignment

Fig.3 depicts the results of relation alignments, showing that AVR-GCN consistently outperforms other methods on all baselines. Only $Hits@1 - Hits@5$ are shown in the graph because the datasets have insufficient numbers of relations tagged. It should be noted that conventional GCNs are not capable of performing relation alignment due to their lack of support for relation embeddings. Therefore, they are not included. In comparison with translation-based models and NTAM, AVR-GCN benefits from its modeling of interactions between entity and relation in the process of convolution.

4.2 Link Prediction

Link prediction, which is widely used to evaluate the effectiveness of network embedding models, does training and testing with a single network. It targets at predicting the missing h or t for a triplet (h, r, t) in a given KG to improve the completeness of the network.

Following R-GCN, we adopt the graph auto-encoder model, which consists of an entity-relation encoder and a scoring function (decoder). VR-GCN acts as an encoder to map each entity and relation to the real-valued vector. For the scoring function needs to support the non-linear and translational properties in convolution process, we utilized $h+r$ and $t-r$ to replace t and h in the scoring function proposed in DistMult [Yang *et al.*, 2014], then:

$$f(h, r, t) = (h + r)M(t - r) \tag{6}$$

The triplets (h, r, t) appeared in the dataset were taken as the positive samples, and negative samples were generated by randomly corrupting each triplet (h, r, t) with head h or tail t replaced. The cross-entropy was utilized to restrict that positive samples have higher scores than negative ones.

Datasets and Baselines. The experiments were based on two well-known benchmark datasets, WN18 and FB15k-237[1] which were extracted from the relational database WordNet and Freebase respectively. The State-of-the-art models included for comparison are TransE, DisMult, and R-GCN.

Result. Table 4 shows the results of link prediction by baselines and VR-GCN on two individual networks. VR-GCN has consistent improvements on both datasets. VR-GCN and R-GCN both perform better than TransE, indicating that deep model has a stronger capability of acquiring the structure information than TransX. VR-GCN outperforms the baseline DistMult, showing the effectiveness of the VR-GCN encoder. VR-GCN also performs better than R-GCN favorably, which highlights the significance of vectorized relation embedding.

5 Conclusion

This paper proposes a vectorized relational graph convolutional network (VR-GCN) to learn the embeddings of both graph entities and relations simultaneously for multi-relational networks. The role discrimination and translation property of knowledge graphs are adopted in the convolutional process. Thereafter, AVR-GCN, the alignment framework based on VR-GCN, is developed for multi-relational network alignment tasks. The weight sharing mechanism is utilized to join the embeddings of graphs into a unified space for alignment. Anchors are used to supervise the objective function which aims at minimizing the distances between anchors. Moreover, anchors are also used to generate cross-network triplets to build a bridge between knowledge graphs at the level of triplet to improve the performance of alignment. The experimental results on the real-world datasets show that the proposed solutions outperform the state-of-the-art methods in network embedding, entity alignment, and relation alignment. For future work, we plan to integrate the attention mechanism and semantic information into the model.

[1]FB15k-237 is a version of FB15k with problematic inverse relation pairs removed.

References

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 2787–2795, 2013.

[Bruna *et al.*, 2013] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *CoRR*, abs/1312.6203, 2013.

[Chen *et al.*, 2017] Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1511–1517, 2017.

[Defferrard *et al.*, 2016] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, Barcelona, Spain*, pages 3837–3845, 2016.

[Grover and Leskovec, 2016] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA*, pages 855–864, 2016.

[Kipf and Welling, 2016] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016.

[Li *et al.*, 2018] Shengnan Li, Xin Li, Rui Ye, Mingzhong Wang, Haiping Su, and Yingzi Ou. Non-translational alignment for multi-relational networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, pages 4180–4186, 2018.

[Lin *et al.*, 2015] Yankai Lin, Zhiyuan Liu, and Maosong Sun. Modeling relation paths for representation learning of knowledge bases. *CoRR*, abs/1506.00379, 2015.

[Liu *et al.*, 2017] Lin Liu, Xin Li, William K. Cheung, and Chengcheng Xu. A structural representation learning for multi-relational networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25*, pages 4047–4053, 2017.

[Niepert *et al.*, 2016] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *Proceedings of the 33nd International Conference on Machine Learning, ICML, New York City, NY, USA*, pages 2014–2023, 2016.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 701–710, 2014.

[Schlichtkrull *et al.*, 2017] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. *CoRR*, abs/1703.06103, 2017.

[Sun *et al.*, 2018] Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. Bootstrapping entity alignment with knowledge graph embedding. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, pages 4396–4402, 2018.

[Velickovic *et al.*, 2017] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *CoRR*, abs/1710.10903, 2017.

[Wang *et al.*, 2014] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Québec City, Québec, Canada*, pages 1112–1119, 2014.

[Wang *et al.*, 2018] Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. Cross-lingual knowledge graph alignment via graph convolutional networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 349–357, 2018.

[Yang *et al.*, 2014] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *CoRR*, abs/1412.6575, 2014.

[Ying *et al.*, 2018] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 974–983, 2018.

[Zhu *et al.*, 2017] Hao Zhu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. Iterative entity alignment via joint knowledge embeddings. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI, Melbourne, Australia*, pages 4258–4264, 2017.