Embedding Knowledge Graphs with Semantic-Guided Walk

Hao Tang, Donghong Liu, Xinhai Xu*, and Feng Zhang*
Academy of Military Science, Beijing, China

***Abstract*—Knowledge graph completion can complete knowledge by predicting missing facts, which is a increasingly hot research topic in knowledge graph construction. Prevalent approaches propose to embed knowledge graphs in a low-dimensional vector space and use these embedding to predict, but they neglect either semantic information or graph structures. We propose a new approach to knowledge graph completion named as ATTWALK, which learns embedding by exploiting both structural and semantic features of a knowledge graph. This is achieved by leveraging a key insight that an entities' embedding is influenced by its multi-hop neighbors', which can be further distinguished by their semantic importance to the entity. ATTWALK orchestrates a two-step workflow by first evaluating neighbors' semantic weights using graph attention networks for each entity, then exploring the entities' local structural features by performing a semantic weight guided walk. We evaluate ATTWALK by conducting extensive experiments, which show that ATTWALK outperforms 12 representative approaches on average across 3 publicly available datasets.**

***Index Terms*—knowledge graph embedding, graph attention networks, random walk**

I. INTRODUCTION

A knowledge graph (KG) is a directed graph which excels at organizing relational facts. It represents factual entities as nodes and semantic relations as edges. For a fact that entity `h` has a relationship `r` with entity `t`, KGs model it as an edge *r* pointing from node *h* to node *t*, which is denoted as a triple (*h*, *r*, *t*). As a structured form to model human knowledge, many large-scaled KGs have emerged as the backbone of many AI related applications such as question answering [1], recommendation systems [2] and intelligent services [3], and become increasingly important nowadays.

Although KGs can be large in size, they are far from complete. This gives rise to the task of automatic KG completion, which aims at predicting missing facts based on existing triples in a KG. A prevalent research direction proposes to map nodes and edges of a KG into distributed representations in a low-dimensional vector space, so as to simplify the prediction while preserving their relations. This is also known as knowledge graph embedding (KGE) and is gaining massive attention recently.

Among all the previous KGE works, facts-based and relation path-based approaches are two main representatives, but they either overlook rich structural features or semantic information of a KG. Facts-based approaches take a KG as a set of triples (i.e., facts). They propose different scoring functions on the embedding of each triple to measure its factual plausibility, and obtain the embeddings by maximizing the function values. Instead, relation path-based approaches compute embeddings considering multi-hop relationships and handle relation paths using composition strategies [4], but the huge number of composed paths brings critical complexity challenges [5]. This gives rise to more feasible approaches of path sampling, which often relies on heuristics or prior knowledge but under-utilizes semantic information in a KG. Such approaches include Node2Vec [6], RelWalk [7] and so on.

Motivated by the above challenges, we propose a new approach named ATTWALK to knowledge graph completion, which learns embedding by exploiting both structural and semantic features of a KG. Unlike using semantic information from textual material [8], we consider using semantic information from the triple itself. ATTWALK is inspired by a point of view in social reference theory [9] that an individual's social role is influenced by his or her personal relational network instead of only direct links, moreover, those entities in this network contribute differently to his or her social role. As an analogy, for a node n in a KG, n's embedding essentially models its role or feature, which is influenced by the *community* (i.e., a sub-graph) c that n is located in, instead of only n's one-hop neighbors, and the influences posed by nodes in c may vary according to their semantic importance to node n. Following this idea, ATTWALK orchestrates a simple two-step workflow: for each node n, its multi-hop neighbors' semantic weights are first evaluated using graph attention network, then the weights are used to guide a random walk starting from n in order to aggregate its multi-hop neighbors' influences and obtain n's embedding.

In summary, the contributions of this paper are threefold: (1) We borrow an idea from sociology and propose a simple but effective approach ATTWALK to KGE. ATTWALK provides a new angle that can exploit both graph structures and semantic relations to embedding KGs; (2) We design and implement a workflow to put the idea of ATTWALK into effects; and (3) We evaluate ATTWALK by conducting extensive experiments. ATTWALK outperforms 12 related approaches on average across three publicly available datasets, which demonstrates the effectiveness of ATTWALK.

II. MOTIVATION

Figure 1 serves as a motivating example throughout this section, which is simplified from a large-scaled KG due to space limit.

DOI reference number: 10.18293/SEKE2022-048
* corresponding author

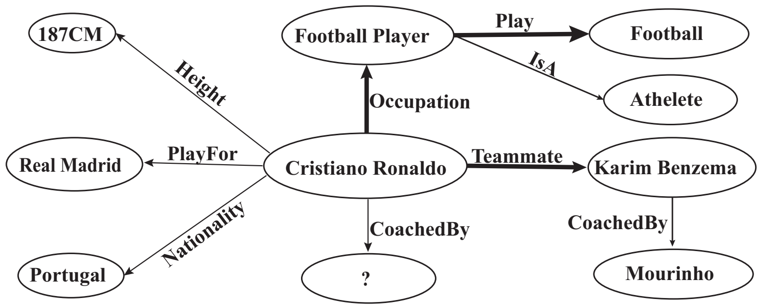

Fig. 1. A KG example describing some real-life relations connected with Cristiano Ronaldo.

We first describe related preliminaries to KGE (Section II-A), then introduce the general ideas of existing works (Section II-B) and our approach (Section II-C) in solving the KGE task. Without loss of generality, let us focus on how these approaches learn the embedding of the specific node *Cristiano Ronaldo* in Figure 1

A. Preliminaries

Knowledge Graph. A knowledge graph $\mathcal{G} = (\mathcal{E}, \mathcal{R})$, where $\mathcal{E}$ and $\mathcal{R}$ represent the set of entities and relations respectively. Each triple (edge) (h, r, t) contains a subject entity $h \in \mathcal{E}$, a predicate $r \in \mathcal{R}$, and a tail entity $t \in \mathcal{E}$, denoting head entity h has a relation r to tail entity t. For example, (*Cristiano Ronaldo*, *Nationality*, *Portugal*) implies the nationality of *Cristiano Ronaldo*, as is shown in Figure 1.

B. Related Work

Facts based approaches generally take the KG in Figure 1 as a set of 8 triples, one for each edge connecting two nodes. For each triple, they propose a model to describe how the triple holds and a scoring function to measure its plausibility. Embeddings are obtained by maximizing the value of the scoring function. In this way, the semantic of each triple is captured, but the multi-hop neighbors' influence and graph structures are not fully utilized. For example, although *Football* is a 2-hop neighbor of *Cristiano Ronaldo*, it is more semantically relevant than *Cristiano Ronaldo*'s 1-hop neighbor *187cm*. These approaches can be divided into four categories: (i) translation-based models, which consider the translation operation between entity and relation embedding, such as TransE [10] and TransH [10]; (ii) factorization-bsaed models, which assume KG as a third-order tensor matrix, and the triple score can be carried out through matrix decomposition. Such as RESCAL [11], HOLE [12]; (iii) CNN-based models, which employ convolutional neural networks to determine the scores of triples, such as ConvE [13] and ConvKB [14]; and (iv) Graph neural network-based models, which extend convolution operations onto non-Euclidean graph structures, such as RGCN [15], KBGAT [16], EIGAT [17] and CompGCN [18].

By contrast, the path sampling based approaches lay more emphasis on the graph structures around the target node. They perform a truncated random walk starting from node *Cristiano Ronaldo* along the outgoing edges, which results in a path p, then propagate embeddings of nodes in p to the target node's, such as DeepWalk [19], node2vec [6]. However, these approaches learn the embedding of *Cristiano Ronaldo* based on its co-occurrence with nodes in the path p, without utilizing the semantics of edges shown in Figure 1. For example, node *187cm* may pose more influence than node *Football Player* on the embedding of node *Cristiano Ronaldo*, though the former is less semantically relevant than the latter.

C. Our Solution

ATTWALK learns the embeddings considering both semantic information and graph structures of the KG, and conducts a two-step procedure. Step-1: For each node n, ATTWALK first evaluates the importance of n's neighbors, based on the intuition that different neighbors pose different influences to n. For example, to node *Cristiano Ronaldo*, the path *Occupation* $\rightarrow$ *Football Player* should contribute more to its embedding than path *Height* $\rightarrow$ *187cm*, therefore, the former is assigned more weights than the latter. This is achieved using a graph attention network. Step-2: Starting from node *Cristiano Ronaldo*, a truncated walk is performed, but which path to take is guided by the weights of the paths. The sampled paths are then aggregated to capture multi-hop neighbors' influences on node *Cristiano Ronaldo*. In this way, both structural and semantic features are exploited in the embedding. Moreover, the relational importance is automatically learned without prior knowledge.

III. ATTWALK

This section introduces our approach ATTWALK. The general architecture is shown as Figure 2. We detail its critical technical components, the graph attention module and the weighted walk module, in Section III-A and Section III-B, respectively.

A. Graph Attention Network

ATTWALK leverages both entity and relation features in the multi-relation knowledge graph. Because in the knowledge graph, we think relations are as crucial as entities. To better manipulate entity and relation embedding, such as inner product, cross product, and subtraction, we map both of them into the same dimension. Each layer of GAT takes a set of entity features

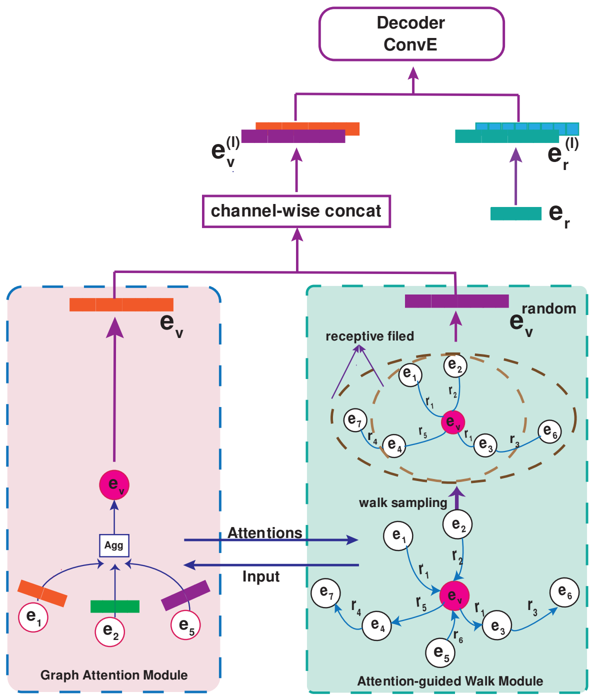

Fig. 2. Overview of AttWalk architecture.

$e \in \mathbb{R}^{N_e \times P}$ and relation features $r \in \mathbb{R}^{N_r \times P}$ as input, and outputs a new set of entity and relation features: $e \in \mathbb{R}^{N_e \times V}$, $r \in \mathbb{R}^{N_r \times V}$, where the i-th row of e is the embedding of entity e_i and the j-th row of r is the embedding of relation r_j. N_e and N_r is the number of entities and relations, respectively. P and V are the input and output dimension of entity and relation embedding, respectively.

Considering different neighbors may have different importance related to one entity, we perform a shared attention mechanism. We first learn a representation of each triple, for example, (h_i, r_j, t_k), by performing a linear transformation over entity and relation embedding, as shown in Equation 1.

$$c_{ijk} = \mathbf{W}\Phi(\vec{h}_i, \vec{r}_j, \vec{t}_k) \tag{1}$$

where Φ represents operations over entity and relation embedding. Inspired from [10], [20] and [21], we define three kinds of operators, subtracting, multiplying and cross-product. $\mathbf{W}$ is the linear transformation matrix. We also notice that knowledge graphs are directed relational graphs. As shown in Figure 1, *Football Player* is embraced by neighborhood entities *Football* and *Cristiano Ronaldo*, and linked by out-relation *Play* and in-relation *Occupation*. *Football Player* can be either a head entity or a tail entity, so we distinguish the direction of the relations. We learn two disjoint patterns of relations, out-relations and in-relations, respectively. Therefore, Equation 1 can again be written as follows.

$$c_{ijk} = \mathbf{W_1}\Phi(\vec{h}_i, \vec{r}_j, \vec{t}_k) \tag{2}$$

$$c_{kji} = \mathbf{W_2}\Phi(\vec{t}_k, \vec{r}_j, \vec{h}_i) \tag{3}$$

where $\mathbf{W_1}$ and $\mathbf{W_2}$ are direction-specific linear transformation matrices. Similar to [16], we learn the importance of each triple denoted by att_{ijk}.

$$att_{ijk} = \text{LeakyReLU}\ (\mathbf{W}_3 c_{ijk}) \tag{4}$$

where $\mathbf{W_3}$ is a linear transformation matrix that is used to calculate attention scores. To get the relative attention values, a softmax function is applied over att_{ijk} as shown in Equation 5.

$$\begin{aligned} \alpha_{ijk} &= \text{softmax}_{jk}\,(att_{ijk}) \\ &= \frac{\exp\,(att_{ijk})}{\sum_{n \in \mathcal{N}_i} \sum_{r \in \mathcal{R}_{in}} \exp\,(att_{inr})} \end{aligned} \tag{5}$$

To aggregate information from neighbor u, the feature of node v is updated by:

$$\vec{e}_v = \sigma_1 \left(\sum_{u \in \mathcal{N}_v} \sum_{i \in \mathcal{R}_{vu}} \alpha_{viu} c_{viu} \right) \tag{6}$$

As is shown in Figure 2, we incorporate the weighted walk model to get the final entity embedding.

$$\vec{e}_v = \sigma_1 \left(\sum_{u \in \mathcal{N}_v} \sum_{i \in \mathcal{R}_{vu}} \alpha_{viu} c_{viu} \right) \| \sigma_1(e_v^{random}) \tag{7}$$

where $\|$ represents channel-wise concatenation, σ_1 is a non-linearity, and e_v^{random} denotes the representation of node *v* derived from the weighted walk model, which will be explained in next section. After updating entity embedding, the relation embedding is transformed by Equation 8.

$$\vec{r}_j = \mathbf{W}_4 r_j \tag{8}$$

where $\mathbf{W}_4$ is a relation transformation matrix that is used to update relation *j*.

B. Weighted Walk

Unlike Node2Vec [6], which views neighbors as equally important, we think different neighbors in a graph play different roles for a specific node, thus have different contributions to the node's embedding. This is achieved by incorporating attention schemes into the random walk process. We propose an attention-guided random walk aggregation model following the hypothesis that accumulating information from local structural relations of n-step ranges will benefit learning robust embedding.

Therefore, instead of feeding adjacent matrix A to implement random walks, we utilize attention matrix D generated from the graph attention module. As is shown in Equation 9, we define linear combinations of features.

$$H^{i+1} = \widehat{D}^j H^i \tag{9}$$

where H^i represents input entity embedding. The weighted walk algorithm is shown as Algorithm 1. The j^{th} line of H^{i+1} is a representation of entity *j*, denoted as e_j^{random}.

We view the random walking process as a Markov chain. Let D denote the one-step transfer probability matrix (attention

Algorithm 1 The weighted walk algorithm

1: **procedure** RANDOMWALK(D^j, H^i)
2: $H^i \leftarrow WH^i + b$
3: $H^i \leftarrow drop(\sigma(H^i))$
4: $\widehat{D}^j = WeightedPruning(D^j)$
5: **for** $i \leftarrow 1, walklength$ **do**
6: $H^{i+1} = \widehat{D}^j H^i$
7: **end for**
8: $H^{i+1} = \sigma(H^{i+1})$
9: **return** H^{i+1}
10: **end procedure**

matrix), whose state representation space is the set of all entities. D has the following properties:

$$d_{ij} \geq 0, i, j \in \mathcal{E} \tag{10}$$

$$\sum_{j \in \mathcal{E}} d_{ij} = 1, i \in \mathcal{E} \tag{11}$$

Based on that, we have the following definition.

Definition III-B.1. Let the conditional probability be defined as $p_{ij}^{(n)} = P\{X_n = j | X_m = i\}, i, j \in \mathcal{E}$, where entity X_n is n-step neighbor of entity X_m.

Lemma III-B.1. The n-step transfer matrix (attention matrix) has the following property:

$$D^{(n)} = DD^{(n-1)} \tag{12}$$

$$D^{(n)} = D^n \tag{13}$$

From the above properties, we can accumulate n-step information by the one-step transfer matrix, i.e., attention matrix. Line 4 in Algorithm 1 tends to select entities of higher importance, that is, we select a local subgraph. For example, as shown in Figure 1, starting from *Cristiano Ronaldo*, the weighted walk process will choose *Football Player* with higher probability over other neighbors. The final subgraph we obtain will be an n-step local structure composed of relatively important entities.

IV. EXPERIMENTS

We first introduce the experiment settings, including the datasets and experiment descriptions (Section IV-A), and the configurations (Section IV-B), then report the overall performance results on KG completion task (Section IV-C), and finally investigate the contributions of different components of ATTWALK by conducting an ablation study (Section IV-D).

A. Datasets

We evaluate our approach on three publicly available benchmark datasets: WN18RR [13], FB15k-237 [26] and Kinship [27]. We use the standard training, validation, and test sets. FB15k-237 contains entities and relations from Freebase, which is a large common-sense knowledge base. FB15k-237 removes duplicate and inverse relationships to prevent direct prediction. WN18RR is derived from WordNet, a lexical database of semantic relations between words. Similar to FB15k-237, WN18RR also removes duplicates and reverse relationships. The Kinship database consists if relationships of 24 unique entities in two families.

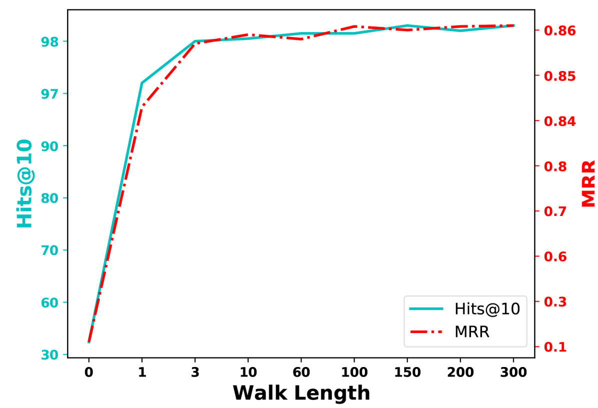

Fig. 3. Impact of walk length on Kinship

B. Configurations

We implement our approach with Pytorch and use Adam to optimize the parameters with an initial learning rate set as 0.001. We run our model under Ubuntu 18.04 on an i9-9900K CPU, equipped with RTX 2080ti 12GB. The embedding size V is set to 200, and the number of negative samples is fixed as 1000. The dropout rate is selected from {0.1, 0.2, 0.3, 0.4}. For algorithm 1, the walk length is tuned amongst {1, 3, 10, 60, 100}. The kernel size of convolution is set as 7×7. We assign label 1 to valid triples and label 0 to negative triples to distinguish them. We use the `CrossEntropyLoss` function as our loss function. Each experiment runs five times and the average number is reported.

C. Results and Analysis

Table I and Table II show the comparison results on all data sets. We can observe that our proposed approach ATTWALK has comparable performance with SOTA baselines on most of the metrics, validating the effectiveness of exploiting both structural and semantic features of a KG. For Kinship, ATTWALK is always the best performer, which outperforms the best baseline by 3.8% on MRR, 2.5% on MR, 6% on Hits@1, 2.3% on Hits@3 and 0.2% on Hits@10 of Kinship, as shown in Table II.

D. Ablation Study

To analyze the effectiveness of each key module in our proposed approach, we investigate an ablation study, which is shown in Table III. In addition, we compare the behavior of our proposed approach when replacing the weighted walk module with NODE2VEC under different aggregation mechanisms and show the results in Table IV. Finally, we compare the effects of different walk lengths, as shown in Figure 3.

1) Effects of Different Modules: As shown in Table III, removing the weighted walk module clearly degrades all the performance metrics, which denotes the effectiveness of weighted walk. When we do not use the updated entity

TABLE I
EXPERIMENTS RESULTS FOR THE LINK PREDICTION TASK ON WN18RR AND FB15K-237 TEST SETS. HITS@N VALUES ARE IN PERCENTAGE. THE BEST SCORE IS IN **BOLD** AND THE SECOND IS UNDERLINED. THE RESULTS OF ALL THE BASELINE METHODS ARE TAKEN FROM THE PREVIOUS PAPERS('-' DENOTES MISSING VALUES).

	WN18RR					**FB15k-237**				
			Hits@N					Hits@N		
	MRR	MR	@1	@3	@10	MRR	MR	@1	@3	@10
TransE [10]	0.226	3384	-	-	50.1	0.294	357	-	-	46.5
DistMult [20]	0.43	5110	39	44	49	0.241	254	15.5	26.3	41.9
ComplEX [22]	0.44	5261	41	46	51	0.247	339	15.8	27.5	42.8
RGCN [15]	-	-	-	-	-	0.248	-	0.151	-	41.7
ConvE [13]	0.43	4187	40	44	52	0.325	244	23.7	35.6	50.1
ConvKB [14]	0.249	3324	5.7	41.7	52.4	0.243	311	15.5	37.1	42.1
KBGAT [16]	0.412	**1921**	-	-	55.4	0.157	270	-	-	33.1
SACN [23]	0.47	-	43	48	54	0.35	-	26	39	54
RotatE [24]	0.476	3340	42.8	<u>49.2</u>	**57.1**	0.338	<u>177</u>	24.1	37.5	53.3
ConvR [25]	0.475	-	<u>44.3</u>	48.9	53.7	0.35	-	26.1	38.5	52.8
CompGCN [18]	<u>0.479</u>	3533	<u>44.3</u>	48.9	53.7	<u>0.355</u>	197	<u>26.4</u>	<u>39.0</u>	<u>53.5</u>
RelWalk [7]	0.451	3232	42	47	51	0.329	**105**	24.3	35.4	50.2
ATTWALK (ours)	**0.483**	<u>2810</u>	**44.5**	**49.7**	<u>56</u>	**0.36**	195.8	**26.8**	**40**	**54.5**

TABLE II
EXPERIMENTS RESULTS FOR THE LINK PREDICTION TASK ON KINSHIP TEST SETS. HITS@N VALUES ARE IN PERCENTAGE. THE BEST SCORE IS IN **BOLD** AND THE SECOND IS UNDERLINED. THE COMPARISONS ARE FROM [16]. WE REPRODUCE THE RESULTS OF KBGAT, RELWALK AND COMPGCN USING [28], [7] AND [18] RESPECTIVELY.

	Kinship				
			Hits@N		
	MRR	MR	@1	@3	@10
TransE [10]	0.309	6.80	0.9	64.3	84.1
DistMult [20]	0.516	5.26	36.7	58.1	86.7
ComplEX [22]	0.823	2.48	73.3	89.9	97.1
RGCN [15]	0.109	25.92	0.3	8.8	23.9
ConvE [13]	0.833	<u>2.00</u>	<u>73.8</u>	<u>91.7</u>	98.1
ConvKB [14]	0.614	3.3	43.62	75.5	95.3
KBGAT [16]	0.548	4.25	36.8	66.5	91.5
CompGCN [18]	<u>0.835</u>	2.06	74.5	91.4	<u>98.3</u>
RelWalk [7]	0.377	4.7	18.4	43.1	92
ATTWALK (ours)	**0.867**	**1.95**	**79.0**	**93.5**	**98.5**

embedding of the graph attention network, but only consider the entity representation obtained from the weighted walk model, we find that experimental results are comparable to ConvE [13], DistMult [20] and RelWalk [7], resulting from that local graph structure features are captured, denoting the effectiveness of Algorithm 1.

2) ATTWALK v.s. Word Embedding Model: From Table IV, we find that combining the graph attention network and the weighted walk model as our encoder provides competitive performance for the ConvE [13] score function. Analyzing the experimental results, TransE [10] does not perform as well

TABLE III
THE EFFECT OF EACH MODULE ON MODEL PERFORMANCE. ATTWALK-W REPRESENTS ATTWALK WITHOUT WEIGHTED WALK MODULE. ATTWALK-O REPRESENTS THERE IS ONLY WEIGHTED WALK MODULE.

	MRR	MR	@1	@3	@10
ATTWALK	**0.483**	**2810**	**44.5**	**49.7**	**56.0**
ATTWALK-W	0.476	3207.4	44.4	49.0	55.0
ATTWALK-O	0.44	4396	40	45.4	52.0

as DistMult [20] and ConvE [13] after the introduction of local structural features, as TransE [10] tends to express the relationships between individual triples, but on the contrary, ConvE [13] has a stronger expression for capturing the relationships between entities and the local structure of entities. In addition, we evaluate ATTWALK by replacing the weighted walk process with word embedding methods and observe a performance decrease, as shown in Table IV.

3) Effects of Walk Lengths: Finally, we evaluate the effects of different walk lengths. As shown in Figure 3, we find that the performance observes an obvious improvement with the increase of walk lengths l at the beginning, but gets stabilized gradually from around $l = 10$. This is in line with our intuition that the more distant an entity is, the smaller its influence is. Besides, note that the training time increases as the walk length increases, so we need to consider performance improvement and time overhead altogether and make a balance.

V. CONCLUSION

In this paper, we propose a new perspective combining graph structures and semantic information in a KG completion

TABLE IV

LINK PREDICTION PERFORMANCE ON KINSHIP DATASET. X+NODE2VEC (Y) INDICATES THAT WE REPLACE THE ATTENTION-GUIDED WALK MODULE, AS SHOWN IN FIGURE 2, WITH THE CLASSICAL WORD EMBEDDING MODEL NODE2VEC, WHERE X IS THE DECODER FUNCTION AND Y IS THE AGGREGATION METHOD.

Decoder→	TransE			DistMult			ConvE		
Methods↓	MRR	MR	@10	MRR	MR	@10	MRR	MR	@10
X+ATTWALK (sub)	0.068	**38.2**	**16.5**	**0.672**	**3.52**	**92.0**	0.669	14.9	78.8
X+ATTWALK (mult)	**0.070**	47.5	11.1	0.612	3.99	90.6	0.841	2.15	97.8
X+ATTWALK (cross)	0.065	47.9	12.9	0.627	3.87	91.0	**0.867**	**1.95**	**98.5**
X+NODE2VEC (sub)	0.06	47.2	11.9	0.587	4.49	89.2	0.385	27.9	50.0
X+NODE2VEC (mult)	0.055	48.3	10.5	0.557	4.68	88.5	0.816	3.35	92.8
X+NODE2VEC (cross)	0.051	48.3	9.5	0.391	7.50	73.8	0.837	2.79	95.8

task. The idea is simple but effective, and can be combined with many existing KGE approaches. Experiments indicate the effectiveness and provide additional insights that the structural expressiveness of random walks can improve the performance of KGE as well as how to set a walk length. We hope our work can shed some light on and inspire more KGE approaches.

ACKNOWLEDGMENT

This work is supported by grants from the National Natural Science Foundation of China (No.11901578 and No.62102444).

REFERENCES

[1] Antoine Bordes, Sumit Chopra, and Jason Weston. Question answering with subgraph embeddings. *arXiv preprint arXiv:1406.3676*, 2014.

[2] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 353–362, 2016.

[3] Yi Ma, Paul A Crook, Ruhi Sarikaya, and Eric Fosler-Lussier. Knowledge graph inference for spoken dialog systems. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5346–5350. IEEE, 2015.

[4] Kelvin Guu, John Miller, and Percy Liang. Traversing knowledge graphs in vector space. In *EMNLP*, 2015.

[5] Shaoxiong Ji, Shirui Pan, E. Cambria, Pekka Marttinen, and Philip S. Yu. A survey on knowledge graphs: Representation, acquisition and applications. *IEEE transactions on neural networks and learning systems*, PP, 2021.

[6] 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*, pages 855–864, 2016.

[7] Danushka Bollegala, Huda Hakami, Yuichi Yoshida, and Ken-ichi Kawarabayashi. Relwalk a latent variable model approach to knowledge graph embedding. *arXiv preprint arXiv:2101.10070*, 2021.

[8] Binling Nie and Shouqian Sun. Knowledge graph embedding via reasoning over entities, relations, and text. *Future Generation Computer Systems*, 91:426–433, 2019.

[9] Pnina Shachaf. Social reference: Toward a unifying theory. *Library & Information Science Research*, 32(1):66–76, 2010.

[10] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013.

[11] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Icml*, 2011.

[12] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. Holographic embeddings of knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[13] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

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

[15] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer, 2018.

[16] Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. Learning attention-based embeddings for relation prediction in knowledge graphs. *arXiv preprint arXiv:1906.01195*, 2019.

[17] Yu Zhao, Huali Feng, Han Zhou, Yanruo Yang, Xingyan Chen, Ruobing Xie, Fuzhen Zhuang, and Qing Li. Eigat: Incorporating global information in local attention for knowledge representation learning. *Knowledge-Based Systems*, page 107909, 2021.

[18] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. Composition-based multi-relational graph convolutional networks. *arXiv preprint arXiv:1911.03082*, 2019.

[19] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[20] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.

[21] Wen Zhang, Bibek Paudel, Wei Zhang, Abraham Bernstein, and Huajun Chen. Interaction embeddings for prediction and explanation in knowledge graphs. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 96–104, 2019.

[22] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *International conference on machine learning*, pages 2071–2080. PMLR, 2016.

[23] Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. End-to-end structure-aware convolutional networks for knowledge base completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3060–3067, 2019.

[24] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*, 2019.

[25] Xiaotian Jiang, Quan Wang, and Bin Wang. Adaptive convolution for multi-relational learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 978–987, 2019.

[26] Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd workshop on continuous vector space models and their compositionality*, pages 57–66, 2015.

[27] Xi Victoria Lin, Richard Socher, and Caiming Xiong. Multi-hop knowledge graph reasoning with reward shaping. *arXiv preprint arXiv:1808.10568*, 2018.

[28] Zhiqing Sun, Shikhar Vashishth, Soumya Sanyal, Partha Talukdar, and Yiming Yang. A re-evaluation of knowledge graph completion methods. *arXiv preprint arXiv:1911.03903*, 2019.