IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 28 March 2023, accepted 2 May 2023, date of publication 15 May 2023, date of current version 24 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3276319

RESEARCH ARTICLE

WeExt: A Framework of Extending Deterministic Knowledge Graph Embedding Models for Embedding Weighted Knowledge Graphs

KONG WEI KUN[1], XIN LIU[2], TEERADAJ RACHARAK[1], (Member, IEEE), GUANQUN SUN[1,3], JIANAN CHEN[1], (Graduate Student Member, IEEE), QIANG MA[4], (Senior Member, IEEE), AND LE-MINH NGUYEN[1]

[1]School of Information Science, Japan Advanced Institute of Science and Technology, Nomi, Ishikawa 923-1292, Japan
[2]Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology (AIST), Tokyo 100-8921, Japan
[3]School of Information Engineering, Hangzhou Medical College, Hangzhou 310053, China
[4]Department of Social Informatics, Kyoto University, Kyoto 606-8501, Japan

Corresponding authors: Kong Wei Kun (kong.diison@jaist.ac.jp); Teeradaj Racharak (racharak@jaist.ac.jp) ; and Le-Minh Nguyen (nguyenml@jaist.ac.jp)

This work was supported in part by the JST SPRING under Grant JPMJSP2102, and in part by the JSPS KAKENHI under Grant JP22K18004.

ABSTRACT With the further development of knowledge graphs, many weighted knowledge graphs (WKGs) have been published and greatly promote various applications. However, current deterministic knowledge graph embedding algorithms cannot encode weighted knowledge graphs well. This paper gives a promising framework *WeExt* that can extend deterministic knowledge graph embedding models to enable them to learn weighted knowledge graph embeddings. In addtion, we introduce weighted link prediction to evaluate the weighted knowledge graph embedding models' performance on completing WKGs. Finally, we give concrete implementation of WeExt based on two translational distance models and two semantic matching models. Our experimental results show the proposed framework achieves promising performance in link prediction, weight prediction, and weighted link prediction.

INDEX TERMS Weighted knowledge graph embedding, weighted link prediction.

I. INTRODUCTION

Knowledge graphs (KG) are thriving and promoting many downstream tasks, such as academic search [1], social relationship recognition [2], and drug discovery [3]. Facts encoded in KG are mostly formalized as triples (h, r, t), in which h denotes the head entity, t denotes the tail entity, and r denotes the relation between h and t. This formalism is sometimes referred to as *deterministic knowledge graph* [4], [5], [6] since triples are employed to represent facts.

Much recent attention has been paid to weighted knowledge graphs (WKG) such as Probase [7], NELL [8], ConceptNet [9], and the Protein-Protein Interaction Knowledge Base STRING [10], [11], which generalize deterministic knowledge graphs by associating a weight $w \in \mathbb{R}$ to each triple. Facts encoded in WKG are mostly formalized as weighted triples $\langle (h, r, t), w \rangle$, though the semantics of weight can be various. For example, it has been used to represent uncertainty [9], confidence score [8], degree of relations [7], edge importance [12], and even out-of-band knowledge [10] in a growing number of scenarios. In real-world usages, it is obvious that the weighted triples model more precise knowledge. For example, while both (*Honda*, *competeswith*, *Toyota*) and (*Honda*, *competeswith*, *Chrysler*) look somewhat correct, the former fact should have a higher confidence than the latter one, since *Honda* and *Toyota* are both Japanese car manufacturers and have highly overlapping customer bases. This modelling can be done if one supposes the semantics of weights based on the confidence score and associates the former triple with a higher value.

As for basic elements in deterministic knowledge graphs, entities and relations are discrete symbols, which are not

The associate editor coordinating the review of this manuscript and approving it for publication was Senthil Kumar.

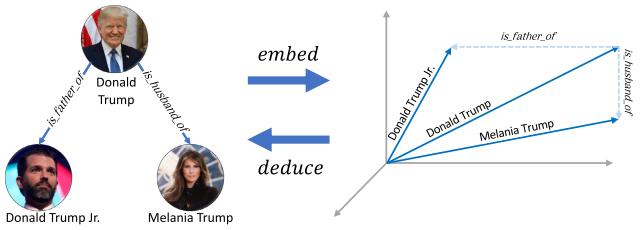


FIGURE 1. Knowledge graph embedding.

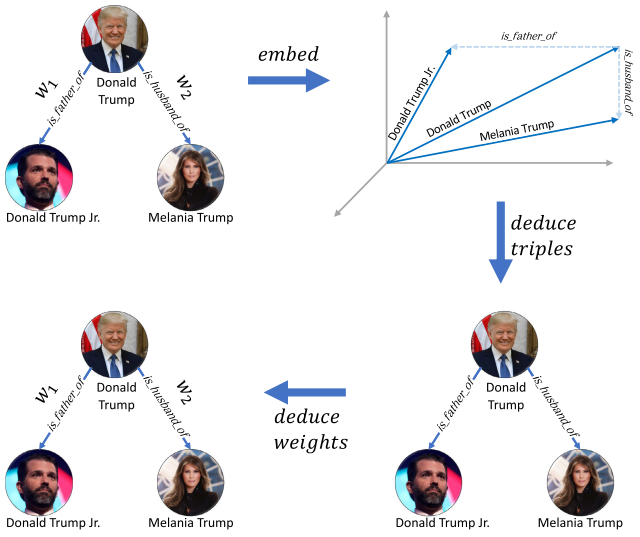


FIGURE 2. Knowledge graph embedding.

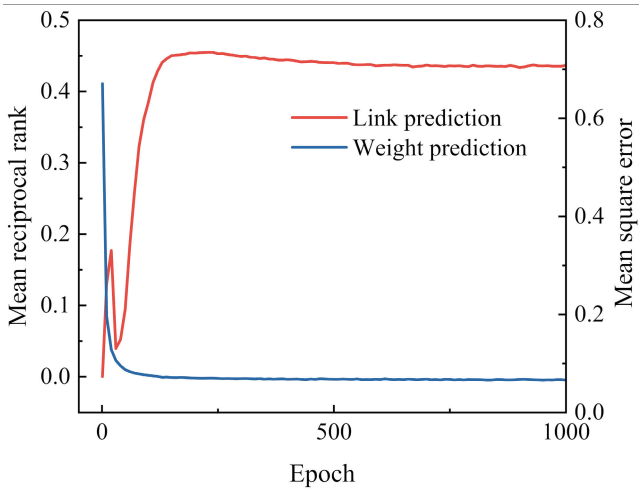


FIGURE 3. The performance of UKGE on NL27K. The red line is the mean reciprocal rank in link prediction. The blue line is the mean square error in weight prediction.

easy to be utilized by machine learning and deep learning models. To address this issue, knowledge graph embedding (KGE) [13] has been investigated to represent the discrete symbols in knowledge graphs as a set of vectors in a specific low-dimensional vector space, and it requires that the representation should enable to deduce the knowledge graphs from this set of vectors. Link prediction (LP) is widely adopted as a task to evaluate the performance of embedding in deducing the structure of any KG. An illustration of knowledge graph embedding is shown in Figure 1.

While KGE algorithms focus on representing deterministic knowledge graphs, they cannot work well when the semantics of triples are imposed by weights. This problem leads to an extended study on weighted knowledge graph embedding (WKGE) aimed to embed entities and relations in a WKG into a set of vectors in a specific low-dimensional vector space. The embeddings of WKG are required to be able to not only deduce the triples in a WKG but also deduce the weight of the triples, which requires that the embeddings of entities and relations have encoded the weight information.

To determine the embedding of weighted triples in WKG, some WKGE algorithms [14], [15] have been proposed to decompose this main task further into two sub-tasks, namely link prediction, and weight prediction. An illustration of this decomposition is shown in Figure 2. However, we observe that this embedding scheme does not achieve the best performance on the link prediction task and on the weight prediction task synchronously. An obvious illustration of the performance of UKGE [14] in link prediction and weight prediction on NL27K is shown in Figure 3. It is easy to notice that while the weight prediction task can perform very well in the early epoch, the performance of the link prediction does not converge to the optimum yet.

There exist works that investigate the encoding weight information in KGE. UKGE [14] and PASSLEAF [15] adopt a non-linear function to convert triple plausibility scores to the weight of the corresponding triples, equating the plausibility of triples with the weight of the triples. But even though the positive triples may have been attached to different weights conveying different meanings, the plausibility of all positive triples should be the same.

To mitigate this issue, we introduce the WeExt framework that uses an independent weight prediction module to existing deterministic knowledge graph embedding models, enabling them to encode the weight information of the triples. The introduced weight prediction module takes the embeddings of any head, relation, and tail entity as input, which contain richer information than the plausibility of the triple. During training, we jointly optimize the model's performance in encoding facts and weights.

We also fill a gap of the WKGE model evaluation task. We design the *weighted link prediction* (WLP) task to comprehensively evaluate the model's ability in deducing weighted triplets. WLP adjusts the ranking of positive triples in all possible triples according to the accuracy of weight prediction to simultaneously demonstrate the performance of the model on link prediction and weight prediction.

We conduct extensive experiments with two representative KGE translational distance models TransE, TransH, and two KGE representative semantic matching models DistMult, HolE. The results show that WeExt achieves competitive

performance over the baseline models on link prediction, weight prediction, and weighted link prediction.

II. RELATED WORK

A. DETERMINISTIC KNOWLEDGE GRAPH EMBEDDING MODELS

Deterministic knowledge graph embedding models [13] are designed for deterministic knowledge graphs, focusing on encoding facts in knowledge graphs. According to different modeling of the interaction between entities and relations, deterministic knowledge graph embedding models can be divided into translational distance models and semantic matching models.

1) TRANSLATIONAL DISTANCE MODELS

The translational distance model, such as TransE [16] and TransH [17], regards the relation as a translation operation from the head entity to the tail entity and utilizes a distance-based scoring function to measure the plausibility of triples.

2) SEMANTIC MATCHING MODELS

The semantic matching models, such as RESCAL [18], DistMult [19], and HolE [20], are based on the tensor factorization and model the interaction of entities and relations by vector-matrix product, obtaining high expressive power due to the use of a full rank matrix for each relation in the score functions which are in the form of $h^\top W_r t$.

B. WEIGHTED KNOWLEDGE GRAPH EMBEDDING MODELS

1) UKGE

UKGE [14] is an embedding model for uncertain knowledge graphs which associate each triple to a confidence score. The model requires logical rules as additional inputs to help enforce the global consistency of predicted facts. UKGE learns the weight of a given triple by squashing the plausibility score of the triple calculated by DistMult using a non-linear function, such as

$$\phi\left(s\left(l\right)\right) = \frac{1}{1 + e^{-(\mathbf{w} \cdot s(l) + \mathbf{b})}} \tag{1}$$

or

$$\phi\left(s\left(l\right)\right) = \min\left(\max\left(\mathbf{w} \cdot s\left(l\right) + \mathbf{b}, 0\right), 1\right). \tag{2}$$

where $\mathbf{w}$ is a weight, $\mathbf{b}$ is a bias and $s\left(l\right)$ is the score of the triple l given by DistMult. UKGE adopts mean square error (MSE) to measure the loss on learning the weights. Given a set of positive relation facts, the loss on the positive triples is

$$\mathcal{L}_{pos} = \sum_{l \in \mathcal{L}^+} \left|\phi\left(s\left(l\right)\right) - w\right|^2 \tag{3}$$

UKGE estimates the weight of negative triples using probabilistic soft logic [21] and measures the loss on negative triples by the square of the distance [22]

$$\mathcal{L}_{neg} = \sum_{l \in \mathcal{L}^-} \sum_{\gamma \in \Gamma} \left|\psi_\gamma\left(\phi(s\left(l\right))\right)\right|^2 \tag{4}$$

where $\mathcal{L}^-$ be a set of negative relaions and Γ be a set of grounded rules. $\psi_\gamma\left(\phi(l)\right)$ denotes the distance to satisfaction of the rule γ in PSL. For any rule $\gamma \equiv \gamma_{body} \rightarrow \gamma_{head}$, the distance d_γ describing the satisfaction of the rule is

$$d_\gamma = \max\left\{0, I\left(\gamma_{body}\right) - I\left(\gamma_{head}\right)\right\} \tag{5}$$

where $I\left(l\right)$ is the soft truth value of the triple l

$$I\left(l\right) = \begin{cases} w, & l \text{ is positive} \\ \phi(s\left(l\right)), & l \text{ is negative} \end{cases} \tag{6}$$

But for negative triples not covered by the rule, the loss is

$$\mathcal{L}_{neg} = \sum_{l \in \mathcal{L}^-} \left|\phi\left(s\left(l\right)\right) - 0\right|^2 \tag{7}$$

which treats the weight of triples not covered by the rules as 0. Thus, the total loss is

$$\mathcal{L} = \mathcal{L}_{pos} + \mathcal{L}_{neg} \tag{8}$$

UKGE only optimizes its ability on weight prediction but does not put attention to its performance on link prediction.

2) PASSLEAF

PASSLEAF [15] extends UKGE to be able to utilize the scoring functions of translation-based approaches and semantic matching-based approaches. PASSLEAF also fits the weight of a given triple using its plausibility and nonlinear function. PASSLEAF utilizes the model being trained itself to predict the weights of unseen triples, rather than utilizing probabilistic soft logic.

Though, as a data augmentation approach, negative sampling can effectively complement the lack of negative triplets in most knowledge graphs, it takes the potential hazard to bring in false-negative samples. PASSLEAF introduces semisupervised samples to ease this issue. The loss function is

$$\mathcal{L} = \mathcal{L}_{pos} + \frac{1}{N_{gen}}\left(\mathcal{L}_{semi} + \mathcal{L}_{neg}\right) \tag{9}$$

where $\mathcal{L}_{pos}$ is the loss on positive triples

$$\mathcal{L}_{pos} = \sum_{l \in \mathcal{L}^+} \left|\phi\left(s\left(l\right)\right) - w\right|^2 \tag{10}$$

$\mathcal{L}_{neg}$ is the loss on negative triples

$$\mathcal{L}_{neg} = \sum_{l \in \mathcal{L}^-} \left|\phi\left(s\left(l\right)\right) - 0\right|^2 \tag{11}$$

$\mathcal{L}_{semi}$ is the loss on negative triples

$$\mathcal{L}_{semi} = \sum_{l \in \mathcal{L}^{semi}} \left|\phi\left(s\left(l\right)\right) - w'\right|^2 \tag{12}$$

PASSLEAF introduces a sample pool to maintain semisupervised samples generated at different time steps to retain experiences accumulated from seeing different randomly drawn negative samples. The pool-based design can be regarded as an ensemble of past models, which further strengthens the effectiveness of semi-supervised samples.

3) TransHExt

TransHExt [23] adopts a 3-layer feed-forward neural network to TransH for predicting the weight w_p of any triple (h, r, t)

$$w_p = \mathcal{N}\left(\left(\mathbf{h} - \mathbf{w}_r^\top \mathbf{h}\mathbf{w}_r\right) + \mathbf{r} - \left(\mathbf{t} - \mathbf{w}_r^\top \mathbf{t}\mathbf{w}_r\right)\right)$$

where $\mathcal{N}$ is the 3-layer feed-forward neural network and $\mathbf{w}_r$ is the normal vector of the relation-specific hyperplane. $\mathbf{h}$, $\mathbf{r}$, $\mathbf{t}$ are corresponding vectors of the triple (h, r, t), respectively. The accuracy of weight prediction is measured by

$$acc(w_p, w) = \begin{cases} \dfrac{w - |w - w_p|}{w}, & w_p \in [0, 2w] \\ 0, & otherwise \end{cases}$$

TransHExt simultaneously optimizes its performance on link prediction and weight prediction through a joint loss

$$\mathcal{L} = \sum_{l \in \mathcal{L}^+} \sum_{l^- \in \mathcal{L}^-} \gamma + \left[f\left(l\right) + acc\left(w_p, w\right)\right] - f\left(l^-\right)$$

C. EVALUATION TASKS FOR KNOWLEDGE GRAPH EMBEDDING MODELS

1) LINK PREDICTION

Link prediction (LP) is the task of predicting the existence of a relation between two entities. LP can be adopted to predict friend relation among users in a social network [24], predict co-author relation in a citation network [25], and predict interactions between genes and proteins in a biological network [26]. Mean rank (MR) [27], mean reciprocal rank (MRR) [28] and Hits@N [16] are widely used for evaluation of the models. For each test triple, the head is removed and replaced by each of the entities of the dictionary in turn. The scores of those corrupted triples are first computed by the models and then sorted by ascending order; the rank of the correct entity is finally stored. This whole procedure is repeated while removing the tail instead of the head. MR calculates the mean of those predicted ranks, MRR calculates the mean of the reciprocal of the ranks, and the Hits@N calculates the proportion of correct entities ranked in the top N.

2) WEIGHT PREDICTION

Weight prediction task (WP) [14] is to predict weights of unseen triples. For each weighted triple $\langle(h, r, t), ?\rangle$ in the test set, the task is to predict the missing weight w. The mean squared error (MSE) and the mean absolute error (MAE) between the predicted values and the ground truth are adopted as the evaluation metrics.

III. METHODOLOGY

A. WeExt

To embed the WKGs, we introduce a weight prediction module consisting of preprocessing and a neural weight predictor (nwp) to predict the weight for a given triple. The architecture of the proposed framework WeExt is shown in Figure 4.

For any deterministic KGE model, a head entity, a relation, and a tail entity interact according to a preset paradigm to produce an interaction vector. This interaction can be divided into two steps, the first step is to preprocess the entities and relations, and the second step is to perform addition operation (translation distance model) or multiplication operation (two-line sex model) to get the interaction vector. The plausibility of the triple is obtained by modulo the interaction vector. UKGE computes the weight of the triple by squashing the plausibility of the triple by a non-linear function, while we argue that the interaction vector of the triple maintains richer information than the plausibility of the triple, thus it may be possible to predict the weights with higher accuracy using the interaction vector.

Based on the above assumption, we design the *inter_vec* (*ivec*) preprocessing. *ivec* processes the entities and the relations following the base model but removes the modulo operation and outputs the interaction vectors directly.

Translational distance models calculate the plausibility of a triple as the negative modulo of the distance between the head and the translated tail entity (as shown in Equation 22 and Equation 23). During training, the model optimizes the distance between the head and the translated tail entity to become smaller. As a result, well-trained translational distance models generate interaction vectors that are close to the zero vector. To mitigate the above problem, we design another preprocessing that concatenates the processed head entity, the relation, and the tail entity after the first step of the interaction of the base model. We call this preprocessing as *concatenating* (*cat*).

We implement WeExt on the basis of four deterministic knowledge graph embedding models, including two representative translational distance models TransE and TransH, and two representative semantic matching models DistMult and HolE.

Next, we demonstrate the implementation of *ivec* and *cat* and how they function. The scoring functions of the base models are presented in Equation 22, Equation 23, Equation 24, and Equation 25, respectively. The *inter_vec* preprocessing for the base models are:

- TransE: $\mathbf{p} = \mathbf{h} + \mathbf{r} - \mathbf{t}$
- TransH: $\mathbf{p} = \left(\mathbf{h} - \mathbf{w}_r^\top \mathbf{h}\mathbf{w}_r\right) + \mathbf{r} - \left(\mathbf{t} - \mathbf{w}_r^\top \mathbf{t}\mathbf{w}_r\right)$
- DistMult: $\mathbf{p} = \mathbf{r} \circ (\mathbf{h} \circ \mathbf{t})$
- HolE: $\mathbf{p} = \mathbf{r} \circ (\mathbf{h} \star \mathbf{t})$

The *concatenating* preprocessings for the base models are:

- TransE: $\mathbf{p} = cat\ (\mathbf{h},\ \mathbf{r},\ -\mathbf{t})$
- TransH: $\mathbf{p} = cat\ \left(\mathbf{h} - \mathbf{w}_r^\top \mathbf{h}\mathbf{w}_r,\ \mathbf{r},\ -(\mathbf{t} - \mathbf{w}_r^\top \mathbf{t}\mathbf{w}_r)\right)$
- DistMult: $\mathbf{p} = cat\ (\mathbf{h},\ \mathbf{r},\ \mathbf{t})$
- HolE: $\mathbf{p} = cat\ (\mathbf{r},\ \mathbf{h} \star \mathbf{t})$

We implement the neural weight predictor using a four-layer feed-forward neural network. The neural weight predictor predicts the weight based on the output of the preprocessing component:

$$w_p = nwp(\mathbf{p})$$

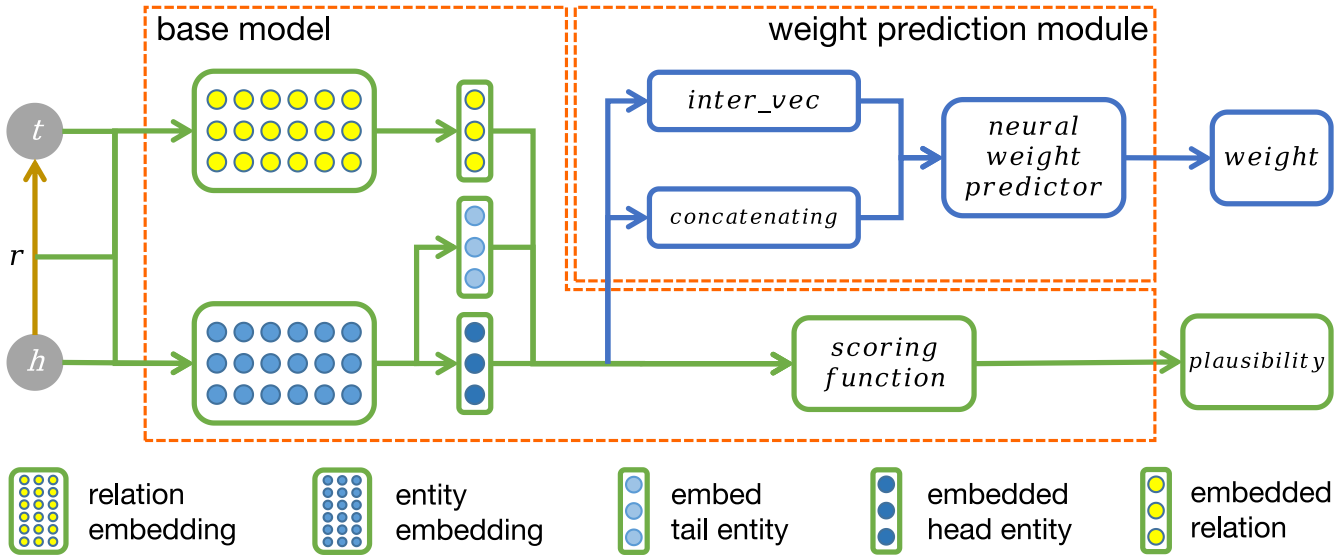


FIGURE 4. The framework of WeExt. The green components are the components of the base KGC model.

To better illustrate the workflow of WeExt, we take TransH as an example and explain how WeExt is used to extend the base model (TransH) in Figure 5.

B. TRAINING PROTOCOL

For a given positive training set

$$S = \left\{\langle (h_i, r_i, t_i), w_i\rangle\right\}_{i=1}^{u},$$

we generate a corresponding negative set by replacing the head entity using all other entities and replacing the tail entity using all other entities:

$$\begin{aligned} S' &= \left\{\left(h', r, t'\right)\right\} \\ &= \left\{\left(h_i', r_i, t_i\right) \mid h_i' \in E \setminus \{h_i\}\right\}_{i=1}^{u} \\ &\quad \cup \left\{\left(h_i, r_i, t_i'\right) \mid t_i' \in E \setminus \{t_i\}\right\}_{i=1}^{u}. \end{aligned} \tag{13}$$

We adopt margin ranking loss [16] to measure the loss on learning the facts:

$$\mathcal{L}_{link} = \sum_{(h,\ell,t)\in S} \sum_{(h',\ell,t')\in S'} \left[\gamma + f(h,r,t) - f\left(h',r,t'\right)\right]_{+} \tag{14}$$

We measure the loss of the weight prediction module on learning the weight of the positive triple using

$$\mathcal{L}_{weight} = \frac{\left|w - w_p\right|}{w} \tag{15}$$

The total loss of the model is

$$\mathcal{L} = (1-\alpha)\cdot \mathcal{L}_{link} + \alpha \cdot \mathcal{L}_{weight} \tag{16}$$

where combination coefficient α is a hyper-parameter that balances the model between learning facts and learning weights.

C. WEIGHTED LINK PREDICTION TASK

1) TASK DESCRIPTION

Weighted link prediction (WLP) aims to simultaneously add missing relations and the corresponding missing weights to the incompleted WKGs. We describe WLP as follows:

Given a weighted knowledge graph

$$WKG = \left\{\langle (h_i, r_i, t_i), w_i\rangle\right\}_{i=1}^{u}$$

where $h_i, t_i \in E$, $r_i \in R$ and $w_i \in (0, 1]$, the E and R are entity and relation sets, respectively. A corrupted weighted triple is defined as a weighted triple without the relation and the weight, i.e., $\langle (h, ?, t), ?\rangle$. WLP is to complete the missing relations and the weights of the corrupted weighted triples in *WKG*, making them to completed weighted triples of the form $\langle (h, r, t), w\rangle$.

2) EVALUATION PROTOCOL

For a test weighted triple $\langle (h_i, r_i, t_i), w_i\rangle$, w_i is omitted. The head entity h_i is replaced by each of the entities of the dictionary in turn to form all possible triples $\langle \left(h_j, r_i, t_i\right), ?\rangle|_{j=1}^{j=|u|}$. Triple scores are calculated by the scoring function of the base model and then sorted in ascending order. After sorting, the ranking of the testing weighted triple rk_i is recorded. This whole procedure is repeated while removing t_i instead of h_i.

We measure the accuracy of predicting the weight by

$$acc(h, r, t, w) = \frac{w - \left|w - w_p\right|}{w} \tag{17}$$

We adjust the ranking of the positive triple rk_i using the accuracy of the weight prediction $acc(h_i, r_i, t_i, w_i)$ and a threshold τ:

$$\mathrm{rk}_i' = \mathrm{rk}_i \cdot exp\left(\tau - acc(h_i, r_i, t_i, w_i)\right) \tag{18}$$

We adopt mean rank (MR), mean reciprocal rank (MRR), and Hits@N to measure the performance of the models on

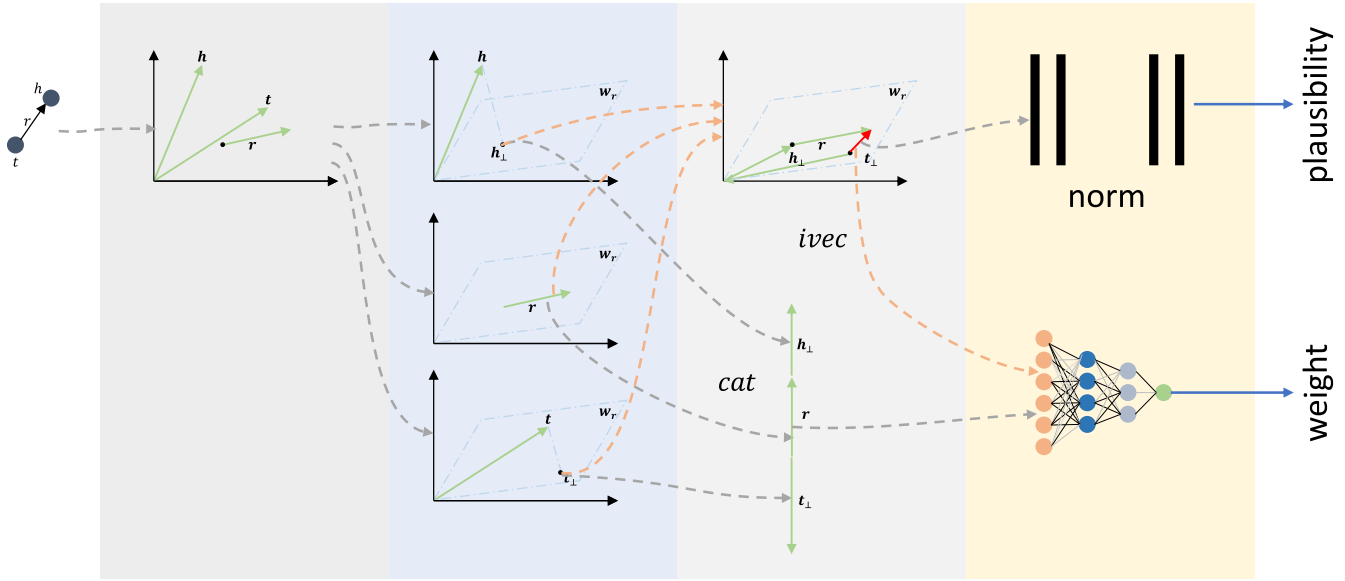


FIGURE 5. An example of the proposed framework based on TransH.

WLP, shown in Equations 19, 20, and 21, respectively.

$$\text{MR} = \sum_{i=1}^{u} \text{rk}'_i, \tag{19}$$

$$\text{MRR} = \sum_{i=1}^{u} \frac{1}{\text{rk}'_i}, \tag{20}$$

$$\text{Hits@N} = \sum_{i=1}^{u} \mathbb{I}[\text{rk}'_i \leq N] \tag{21}$$

where the $\mathbb{I}[expn]$ is the indicator function, which outputs 1 if $expn$ is true, and 0 otherwise.

IV. EXPERIMENTS AND RESULTS

To measure the performance of the proposed WeExt framework, we evaluate the weighted extensions of the base models on link prediction, weight prediction, and weighted link prediction.

A. EXPERIMENT SETTING

We conducted experiments on CN15K, NL27K, and PPI5K [29] datasets. CN15K is a subgraph of ConceptNet [9], containing 15,000 entities and 229,235 weighted triples in English. The original scores in ConceptNet vary from 0.1 to 22, while the weights in CN15K are normalized to [0.1, 1.0]. NL27k is extracted from NELL [8], a weighted KG obtained from webpage reading. NL27k contains 27,221 entities, 405 relations, and 175,412 weighted triples. The weights in NL27K are normalized to the interval [0.1, 1.0]. PPI5k is a subset of the protein-protein interaction knowledge base STRING [10] that contains 255,114 weighted triples for 4,999 proteins and 7 interactions. STRING labels the interactions between proteins with the probabilities of occurrence. The weights in PPI5k fall in the interval [0.15, 1.0]. We drop out duplicated quadruplets in CN15K and PPI5K. The statistics of the WKGs are shown in Table 1.

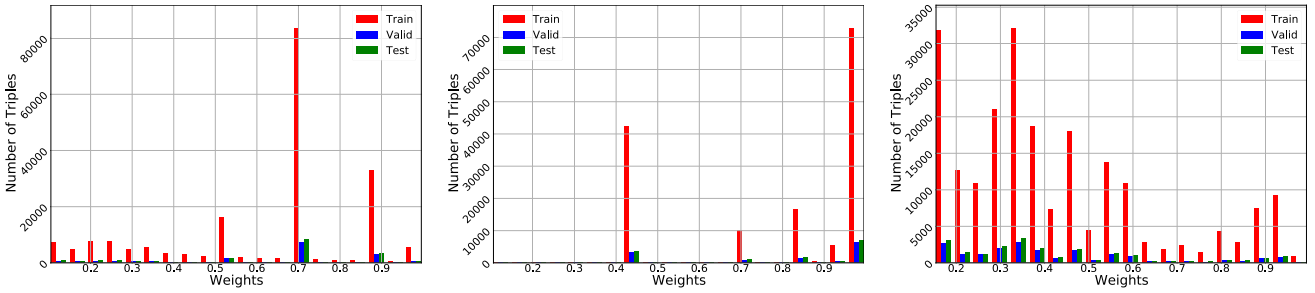
TABLE 1. Statistics of weighted knowledge graphs. #Ent denotes the number of the entities, #Rel denotes the number of the relations, #Tri denotes the number of the triples, INR denotes the interval of the weights, Avg(d) denotes the average of the degree of the entities, and Med(d) denotes the median of the degree of the entities.

		#Ent	#Rel	#Tri	INR	Avg(d)	Med(d)
CN15K	train	15000	36	193274	0.900	25.77	12
	test	10659	34	19166	0.900	3.60	2
	val	10158	35	16795	0.900	3.31	2
NL27K	train	27221	405	149100	0.899	10.95	4
	test	9711	287	14034	0.898	2.89	1
	val	9000	279	12278	0.899	2.73	1
PPI5K	train	4999	7	214661	0.847	85.88	21
	test	3703	7	21566	0.847	11.65	4
	val	3557	7	18887	0.847	10.62	3

We implemented the proposed framework and the weighted link prediction task based on the PyKEEN toolkit [30]. We choose 0.01 as the learning rate λ for the stochastic gradient descent and searched the combination coefficient α for loss function among {0.1, 0.2, 0.01, 0.001, 0.0001}. The margin of the loss function γ was set to 1. The dimension of embeddings was set to 50. We trained the models for 3000 epochs, evaluated the models per 10 epochs, and save their best results.

B. BASE MODELS

We implemented the proposed framework based on two representative translational distance models: TransE and TransH, and two representative semantic matching models: DistMult and HolE.



(a) CN15K. (b) NL27K. (c) PPI5K.

FIGURE 6. The weight distribution in the datasets.

1) TransE

TransE [16] is one of the most representative translational distance models. It interprets entities as vectors and the relation as a translation vector of the head entity in one embedding space. The scoring function of TransE is

$$s = -\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_p \tag{22}$$

2) TransH

It has been well investigated that TransE is effective for 1-to-1 relations, but cannot model the 1-to-N or N-to-N relation well. TransH [17] models 1-to-N and N-to-N relations by introducing the mechanism of projecting to relation-specific hyperplanes. The scoring function of TransH is

$$s = -\left\|\left(\mathbf{h} - \mathbf{w}_r^\top \mathbf{h}\mathbf{w}_r\right) + \mathbf{r} - \left(\mathbf{t} - \mathbf{w}_r^\top \mathbf{t}\mathbf{w}_r\right)\right\|_2^2 \tag{23}$$

where $\mathbf{w}_r$ stands for the normal vector of the relation-specific hyperplane.

3) DistMult

DistMult [19] represents each relation as a diagonal matrix that models pairwise interactions between entities to capture the latent semantics. The scoring function of DistMult is

$$s = \|\mathbf{r} \circ \mathbf{h} \circ \mathbf{t}\| \tag{24}$$

where $\circ$ is the element-wise product.

4) HolE

For a given triple, HolE [20] first composes the head entity and tail entity using the circular correlation operation [31], then matches the relational with the compositional vector of the head entity and tail entity to score the given triple. Since circular correlation is not commutative, HolE is able to model asymmetric relations. The scoring function of HolE is

$$s = \mathbf{r}^\top (\mathbf{h} \star \mathbf{t}) \tag{25}$$

where $\star$ is the circular correlation.

TABLE 2. Results on link prediction.

	Model		MR	MRR	Hits@1	Hits@3	Hits@5	Hits@10
CN15K	UKGE		1760.4	0.0800	0.0372	0.0840	0.1161	0.1634
	PL	TE	4934.5	0.0026	0.0003	0.0010	0.0017	0.0036
		DM	2206.0	**0.1149**	**0.0888**	0.1145	0.1320	0.1619
	TransE		1240.2	0.1078	0.0371	0.1327	0.1818	0.2457
	TE_{Ext}	ivec	1206.4	0.1091	0.037	**0.1357**	**0.1821**	**0.2498**
		cat	1233.3	0.1084	0.0372	0.1324	0.1817	0.2466
	TransH		1716.3	0.0789	0.0419	0.0879	0.1107	0.1455
	TH_{Ext}	ivec	1747.5	0.079	0.0415	0.0898	0.1118	0.1442
		cat	1745.1	0.0804	0.0425	0.0899	0.1132	0.1486
	DistMult		**966.1**	0.1072	0.041	0.13	0.1711	0.2315
	DM_{Ext}	ivec	1105.7	0.0907	0.0408	0.1012	0.1339	0.1831
		cat	1208.5	0.0773	0.0373	0.0815	0.108	0.152
	HolE		1319.7	0.0935	0.0477	0.0994	0.1335	0.1832
	HE_{Ext}	ivec	1327.7	0.0946	0.0496	0.1014	0.1329	0.1813
		cat	1330.9	0.0945	0.0506	0.1007	0.1318	0.1794
NL27	UKGE		236.6	0.4550	0.3697	0.4879	0.5483	0.6268
	PL	TE	11216.9	0.0089	0.0033	0.0092	0.0125	0.0203
		DM	1531.4	**0.5998**	**0.5926**	**0.5927**	0.5932	0.6014
	TransE		111.4	0.3736	0.2298	0.4554	0.5325	0.6344
	TE_{Ext}	ivec	114.8	0.3794	0.2366	0.4593	0.5364	0.6387
		cat	**109.2**	0.3818	0.2344	0.4662	0.5449	0.6465
	TransH		686.1	0.2721	0.1874	0.3125	0.3593	0.42
	TH_{Ext}	ivec	671.2	0.2798	0.1978	0.3168	0.3623	0.4226
		cat	677.5	0.2769	0.194	0.3156	0.3579	0.4193
	DistMult		179.1	0.3993	0.3065	0.4421	0.4939	0.57
	DM_{Ext}	ivec	264.8	0.3703	0.2822	0.4069	0.4573	0.5321
		cat	186.3	0.3729	0.2823	0.4118	0.4634	0.5354
	HolE		135	0.5223	0.4198	0.5716	0.639	0.7241
	HE_{Ext}	ivec	144.5	0.5382	0.4377	0.5873	**0.6507**	**0.7333**
		cat	134.8	0.5291	0.4277	0.579	0.641	0.7267
PPI5K	UKGE		29.3	0.3759	0.2405	0.4320	0.5092	0.6435
	PL	TE	3782.5	0.0065	0.0051	0.0071	0.0074	0.0078
		DM	12.1	0.6557	0.5091	0.7464	0.8481	0.9386
	TransE		19.6	0.1819	0.0001	0.2432	0.3704	0.5636
	TE_{Ext}	ivec	18.4	0.1815	0	0.2397	0.3682	0.5635
		cat	18.7	0.1818	0	0.2425	0.3679	0.5637
	TransH		49.5	0.1106	0.0029	0.1274	0.1906	0.3155
	TH_{Ext}	ivec	50.1	0.1196	0	0.1523	0.2193	0.3334
		cat	48.7	0.1199	0	0.1532	0.217	0.3283
	DistMult		24.1	0.459	0.3427	0.5175	0.5763	0.6662
	DM_{Ext}	ivec	34.6	0.4302	0.3266	0.473	0.5298	0.6186
		cat	32.6	0.4585	0.3549	0.4994	0.5584	0.6495
	HolE		6.6	0.8426	0.7589	0.9149	0.9473	0.9719
	HE_{Ext}	ivec	**6.1**	0.845	0.7628	0.9168	0.9479	0.9722
		cat	6.6	**0.8542**	**0.7762**	**0.9218**	**0.951**	**0.9743**

* PL: PASSLEAF, TE: TransE, TH: TransH, DM: DistMult, HE: HolE.

C. RESULTS ON LINK PREDICTION

We describe the link prediction task and its evaluation protocol in Section II-C1. The results on link prediction are shown in Table 2. Because MRR is not sensitive to extremely

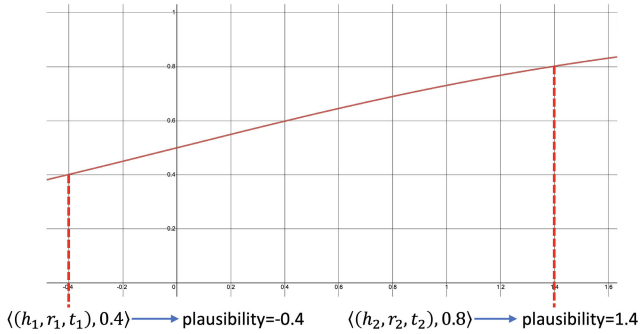


FIGURE 7. An illustration of how UKGE infers weights from the plausibility of triples. Given two triples A = $\langle (h_1, r_1, t_1), 0.4 \rangle$ and B = $\langle (h_2, r_2, t_2), 0.8 \rangle$, the non-linear function is sigmoid function: $s(t) = \frac{1}{1+e^{-1}}$. Let a well-trained UKGE model predict the plausibility of the given triples, the plausibility of triple-A will be −0.4 and plausibility of triple-B will be 1.4.

poor rankings, we mainly focus on the MRR score. After introducing a weight prediction module for base models, the extended models have to optimize their performance on both link prediction and weight prediction simultaneously. The results show that the introduced additional optimization term does not cause the model's performance to decrease on the link prediction task; there may be a slight improvement for some models instead, for example, TransE, TransH, and HolE. DistMult$_{Ext}$ achieves a worse performance than DistMult on all the datasets. DistMult$_{Ext}$ outperforms UKGE on CN15K and PPI5K, but due to the improved performance of UKGE on NL27K compared to DistMult, the performance of UKGE on NL27K is much better than DistMult$_{Ext}$.

Due to the sample pool used in PASSLEAF with weight estimation of negative samples, PASSLEAF with DistMult outperforms DistMult, UKGE, and DistMult$_{Ext}$ on CN15K, NL27K, and PPI5K datasets. However, the performance of PASSLEAF is unstable, i.e., PASSLEAF with TransE resulted in the worst performance on all three datasets.

From another perspective, all the extended models outperform UKGE on CN15K. DistMult$_{Ext}$ and HolE$_{Ext}$ can surpass UKGE on PPI5K. But as for NL27K, only HolE$_{Ext}$ achieves better performance than UKGE. We assume the performance difference on the different datasets is caused by the weight distribution of the respective datasets, shown in Figure 6, and the way how UKGE learns the weights. Triples in NL27K are centralized in high-weight regions, triples in PPI5K are centralized in low-weight regions, and weights in CN15K are not so polarized as NL27K and PPI5K.

UKGE utilizes a non-linear function to squeeze the plausibility of triples to obtain the weights of the triples, which makes the triples with a small weight gap to gain a larger plausibility gap. An illustration is shown in Figure 7. Thus, UKGE tends to assign stronger plausibility for high-weight triples and weaker plausibility for low-weight triples, which is why UKGE's performance on PPI5K is not as good as half of the best-performing model. In contrast, on NL27K,

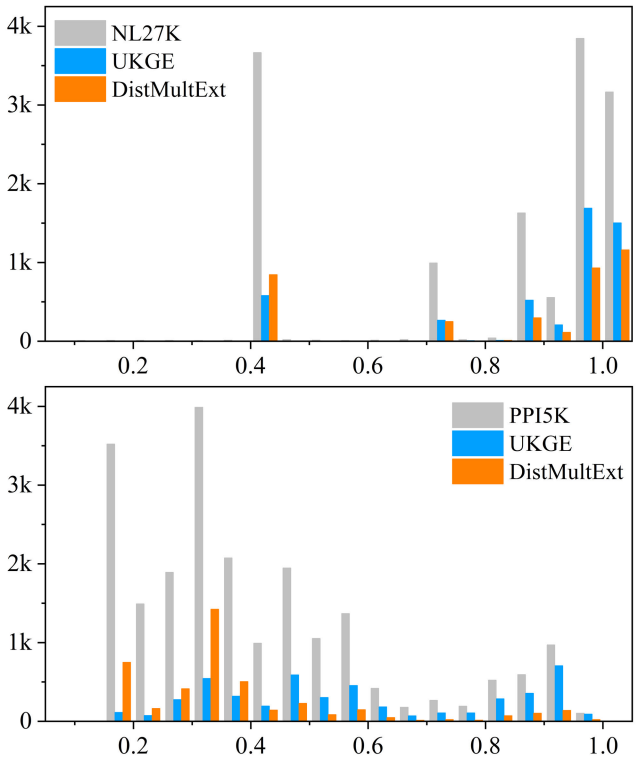


FIGURE 8. An illustration of the weight distribution of triples correctly predicted by UKGE and DistMult$_{Ext}$ on NL27K and PPI5K.

where high-weight triples account for a large proportion, the gap between the performance of UKGE and the best-performing model is much smaller.

Figure 8 shows the weight distribution of triples correctly predicted by UKGE and DistMult$_{Ext}$ on NL27K and PPI5K, we can see that UKGE predicts more triples with high weights, while DistMult$_{Ext}$ performs more balanced on different intervals.

Moreover, WeExt with the *cat* preprocessing outperforms WeExt with the *ivec* preprocessing on PPI5K, and they achieve similar performance on CN15K and NL27K.

D. RESULTS ON WEIGHT PREDICTION

For each weighted triple $\langle (h_i, r_i, t_i), w_i \rangle$ in the test set, we predict the weight based on the triple (h_i, r_i, t_i) and report the mean squared error (MSE) and mean absolute error (MAE).

The results on weight prediction are shown in Table 3. Except for HolE$_{Ext}$ that performs worse than UKGE on NL27K, all the weighted extensions outperform UKGE in the weight prediction task for all three datasets. Compared with UKGE and DistMult$_{Ext}$, PASSLEAF's performance in the weight prediction task is poorer on all datasets. It may be caused by noise introduced during the weight estimation of negative samples by the model being trained. The result shows that adopting neural networks to learn weights from processed embeddings is superior to utilizing non-linear

TABLE 3. Results on weight prediction.

	Model		MSE ($\times 10^{-2}$)	MAE ($\times 10^{-2}$)
CN15K	UKGE		8.61	19.90
	PL	TE	57.91	39.10
		DM	41.66	24.28
	TransE_{Ext}	ivec	4.60	14.66
		cat	3.83	13.08
	TransH_{Ext}	ivec	4.26	13.89
		cat	3.91	12.89
	DistMult_{Ext}	ivec	5.10	15.62
		cat	**3.67**	**11.93**
	HolE_{Ext}	ivec	5.86	17.09
		cat	5.79	16.31
NL27K	UKGE		2.36	6.90
	PL	TE	73.66	59.91
		DM	18.25	8.96
	TransE_{Ext}	ivec	1.45	5.98
		cat	1.23	5.24
	TransH_{Ext}	ivec	2.12	8.12
		cat	3.05	10.46
	DistMult_{Ext}	ivec	1.39	6.13
		cat	**1.22**	5.09
	HolE_{Ext}	ivec	1.31	**5.01**
		cat	1.26	5.38
PPI5K	UKGE		0.95	3.79
	PL	TE	40.74	21.14
		DM	47.53	26.71
	TransE_{Ext}	ivec	0.24	2.77
		cat	0.24	2.58
	TransH_{Ext}	ivec	0.49	3.7
		cat	0.42	3.32
	DistMult_{Ext}	ivec	0.34	2.89
		cat	0.28	2.70
	HolE_{Ext}	ivec	0.16	**1.83**
		cat	**0.15**	1.85

* PL: PASSLEAF, TE: TransE, TH: TransH, DM: DistMult, HE: HolE.

TABLE 4. Results on weighted link prediction.

	Model		MR	MRR	Hits@1	Hits@3	Hits@5	Hits@10
CN15K	UKGE		4031.2	0.0795	0.0266	0.0736	0.1066	0.1556
	PL	TE	10560.9	0.0015	0	0.0003	0.0009	0.0019
		DM	14401.8	0.0776	0.0198	0.0797	0.1040	0.1365
	TE_{Ext}	ivec	3602	0.1129	0.035	0.1153	0.1639	0.2295
		cat	4467.6	**0.1154**	0.0342	**0.1174**	**0.1694**	**0.2361**
	TH_{Ext}	ivec	4907.8	0.0865	0.0393	0.0821	0.1059	0.1398
		cat	4551.3	0.0868	0.0381	0.0829	0.1085	0.1438
	DM_{Ext}	ivec	3071.9	0.0961	0.0365	0.0898	0.1242	0.1749
		cat	3224.2	0.083	0.033	0.0737	0.1025	0.147
	HE_{Ext}	ivec	**2761.2**	0.0984	0.039	0.0924	0.1263	0.1777
		cat	2786.5	0.0992	**0.0433**	0.0886	0.1201	0.1696
NL27K	UKGE		483.5	0.5269	0.3566	0.4844	0.5521	0.637
	PL	TE	24246.2	0.0046	0	0.0034	0.0071	0.0120
		DM	3415.8	0.5861	0.4265	0.5904	0.5920	0.6027
	TE_{Ext}	ivec	**137.1**	0.4198	0.1967	0.4481	0.5356	0.6491
		cat	149.6	0.4093	0.1801	0.4429	0.5378	0.6462
	TH_{Ext}	ivec	621.5	0.2996	0.156	0.3044	0.3547	0.4258
		cat	633.8	0.2942	0.1455	0.3039	0.3544	0.4221
	DM_{Ext}	ivec	280.7	0.4148	0.2518	0.4012	0.456	0.536
		cat	382	0.419	0.2558	0.3996	0.4581	0.5438
	HE_{Ext}	ivec	272	**0.624**	**0.4128**	**0.58**	**0.6563**	**0.746**
		cat	339.1	0.6038	0.3941	0.5579	0.636	0.7236
PPI5K	UKGE		34.3	0.4212	0.2131	0.4193	0.5073	0.6518
	PL	TE	8403.4	0.0047	0.0003	0.0058	0.0070	0.0075
		DM	79.6	0.2378	0.0366	0.2603	0.4376	0.5728
	TE_{Ext}	ivec	18.6	0.1996	0	0.2227	0.3574	0.5631
		cat	18.3	0.2018	0	0.225	0.3697	0.5772
	TH_{Ext}	ivec	47.4	0.1336	0	0.1459	0.2174	0.3441
		cat	45.3	0.1363	0	0.147	0.217	0.345
	DM_{Ext}	ivec	34.5	0.4782	0.2835	0.4641	0.5263	0.6235
		cat	33.7	0.5124	0.309	0.4906	0.5563	0.6547
	HE_{Ext}	ivec	**7.7**	0.9769	0.7241	0.9096	0.9457	**0.9722**
		cat	22.6	**0.9948**	**0.7448**	**0.914**	**0.9473**	0.9717

* PL: PASSLEAF, TE: TransE, TH: TransH, DM: DistMult, HE: HolE.

functions to learn weights by squeezing the plausibility of the triple.

Moreover, WeExt with the *cat* preprocessing outperforms WeExt with the *ivec* preprocessing, not only for the translational distance models but also for the semantic matching models, indicating that after the model is well-trained, the cascade of entities and relations retains richer information than the interaction vector.

E. RESULTS ON WEIGHTED LINK PREDICTION

The results on weighted link prediction are shown in Table 4. All weighted extensions outperform UKGE in weighted link prediction on CN15K. DistMult_{Ext} and HolE_{Ext} outperform UKGE on both NL27K and PPI5K. Although the link prediction performance of DistMult_{Ext} on NL27K is worse than UKGE, DistMult_{Ext} achieves better performance in the weighted link prediction on NL27K thanks to the better weight prediction performance.

While PASSLEAF performs well in link prediction tasks, it falls short of surpassing UKGE and DistMult_{Ext} on CN15K and PPI5K datasets. On the NL27K dataset, although PASSLEAF used with DistMult still outperforms UKGE and HolE_{Ext}, it has been outperformed by HolE_{Ext}. This implies that using the model being trained to estimate the weights of negative samples is beneficial for pure link prediction tasks, but it may introduce noise for tasks that involve weights.

Moreover, WeExt with the *cat* preprocessing outperforms WeExt with the *ivec* preprocessing on PPI5K, but they achieve similar performance on CN15K and NL27K. The performance of the weighted extensions on weighted link prediction is consistent with the performance trend on link prediction, but not consistent with the performance of the model's weighted prediction. This indicates that under the current evaluation protocol for weighted link prediction, the performance of the model on the link prediction task is dominant, while the performance of the model on the weight prediction task has been taken into consideration, but only in a subordinate position.

V. CONCLUSION AND FUTURE WORK

In this paper, we propose a framework called WeExt for extending deterministic knowledge graphs to be capable of embedding weighted knowledge graphs. To facilitate the performance evaluation of our extended WKGE models, we propose the novel weighted link prediction task. Compared with the widely-used asynchronous link prediction and weight prediction tasks, weighted link prediction can synchronously evaluate the performance of weighted knowledge graph embedding in link prediction and weight prediction.

In the next work, we plan to design a new evaluation protocol to alleviate the impact of extreme data on the score, so that the score can better reflect the model performance.

REFERENCES

[1] C. Xiong, R. Power, and J. Callan, "Explicit semantic ranking for academic search via knowledge graph embedding," in *Proc. WWW*, 2017, pp. 1271–1279.

[2] Z. Wang, T. Chen, J. S. J. Ren, W. Yu, H. Cheng, and L. Lin, "Deep reasoning with knowledge graph for social relationship understanding," in *Proc. IJCAI*, 2018, pp. 1–8.

[3] X. Lin, Z. Quan, Z.-J. Wang, T. Ma, and X. Zeng, "KGNN: Knowledge graph neural network for drug-drug interaction prediction," in *Proc. 9th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 2739–2745.

[4] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Jun. 2008, pp. 1247–1250.

[5] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, "DBpedia—A large-scale, multilingual knowledge base extracted from Wikipedia," *Semantic Web*, vol. 6, no. 2, pp. 167–195, 2015.

[6] F. Mahdisoltani, J. Biega, and F. Suchanek, "YAGO3: A knowledge base from multilingual Wikipedias," in *Proc. 7th biennial Conf. Innov. Data Syst. Res.*, 2014, pp. 1–11.

[7] W. Wu, H. Li, H. Wang, and K. Q. Zhu, "Probase: A probabilistic taxonomy for text understanding," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, May 2012, pp. 481–492.

[8] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, B. Yang, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, and J. Krishnamurthy, "Never-ending learning," *Commun. ACM*, vol. 61, no. 5, pp. 103–115, 2018.

[9] R. Speer, J. Chin, and C. Havasi, "ConceptNet 5.5: An open multilingual graph of general knowledge," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 1–8.

[10] D. Szklarczyk, A. Franceschini, S. Wyder, K. Forslund, D. Heller, J. Huerta-Cepas, M. Simonovic, A. Roth, A. Santos, K. P. Tsafou, M. Kuhn, P. Bork, L. J. Jensen, and C. von Mering, "STRING V10: Protein–protein interaction networks, integrated over the tree of life," *Nucleic Acids Res.*, vol. 43, no. D1, pp. D447–D452, Jan. 2015.

[11] D. Szklarczyk, J. H. Morris, H. Cook, M. Kuhn, S. Wyder, M. Simonovic, A. Santos, N. T. Doncheva, A. Roth, P. Bork, L. J. Jensen, and C. von Mering, "The STRING database in 2017: Quality-controlled protein–protein association networks, made broadly accessible," *Nucleic Acids Res.*, vol. 45, no. D1, pp. D362–D368, Jan. 2017.

[12] O. De la Cruz Cabrera, M. Matar, and L. Reichel, "Edge importance in a network via line graphs and the matrix exponential," *Numer. Algorithms*, vol. 83, no. 2, pp. 807–832, Feb. 2020.

[13] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 12, pp. 2724–2743, Dec. 2017.

[14] X. Chen, M. Chen, W. Shi, Y. Sun, and C. Zaniolo, "Embedding uncertain knowledge graphs," in *Proc. AAAI*, 2019, pp. 1–8.

[15] Z. Chen, M.-Y. Yeh, and T.-W. Kuo, "PASSLEAF: A pool-based semi-supervised learning framework for uncertain knowledge graph embedding," in *Proc. AAAI*, 2021, pp. 1–8.

[16] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. NIPS*, 2013, pp. 1–9.

[17] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," *Proc. AAAI Conf. Artif. Intell.*, vol. 28, no. 1, Jun. 2014, pp. 1–8.

[18] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proc. ICML*, 2011, pp. 1–8.

[19] B. Yang, S. W.-T. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," in *Proc. ICLR*, 2015, pp. 1–12.

[20] M. Nickel, L. Rosasco, and T. A. Poggio, "Holographic embeddings of knowledge graphs," in *Proc. AAAI*, 2016, pp. 1955–1961.

[21] A. Kimmig, S. H. Bach, M. Broecheler, B. Huang, and L. Getoor, "A short introduction to probabilistic soft logic," in *Proc. NIPS*, 2012, pp. 1–4.

[22] S. Bach, B. Huang, B. London, and L. Getoor, "Hinge-loss Markov random fields: Convex inference for structured prediction," in *Proc. UAI*, 2013, pp. 1–10.

[23] W. K. Kong, X. Liu, T. Racharak, and L.-M. Nguyen, "TransHExt: A weighted extension for transh on weighted knowledge graph embedding," in *Proc. ISWC*, 2022, pp. 1–5.

[24] L. A. Adamic and E. Adar, "Friends and neighbors on the web," *Social Netw.*, vol. 25, no. 3, pp. 211–230, Jul. 2003.

[25] H. Cho and Y. Yu, "Link prediction for interdisciplinary collaboration via co-authorship network," *Social Netw. Anal. Mining*, vol. 8, no. 1, pp. 1–12, Dec. 2018.

[26] E. M. Airoldi, D. M. Blei, S. E. Fienberg, E. P. Xing, and T. Jaakkola, "Mixed membership stochastic block models for relational data with application to protein–protein interactions," in *Proc. Int. biometrics Soc. Annu. meeting*, vol. 15, 2006, p. 1.

[27] A. Bordes, J. Weston, R. Collobert, and Y. Bengio, "Learning structured embeddings of knowledge bases," *Proc. AAAI Conf. Artif. Intell.*, vol. 25, no. 1, pp. 301–306, Aug. 2011.

[28] E. M. Voorhees, "The TREC-8 question answering track," *Natural Lang. Eng.*, vol. 7, pp. 361–378, Dec. 2000.

[29] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2D knowledge graph embeddings," in *Proc. AAAI*, 2018, pp. 1811–1818.

[30] M. Ali, M. Berrendorf, C. T. Hoyt, L. Vermue, S. Sharifzadeh, V. Tresp, and J. Lehmann, "PyKEEN1.0: A Python library for training and evaluating knowledge graph embeddings," *J. Mach. Learn. Res.*, vol. 22, no. 82, pp. 1–6, 2021.

[31] T. A. Plate, "Holographic reduced representations," *IEEE Trans. Neural Netw.*, vol. 6, no. 3, pp. 623–641, May 1995.

KONG WEI KUN received the joint master's degree from the College of Intelligence and Computing, Tianjin University, and the Department of Information Science, Japan Advanced Institute of Science and Technology, in 2020. He is currently pursuing the Ph.D. degree with the Japan Advanced Institute of Science and Technology. His research interests include natural language processing, knowledge graph representation learning, and touring route recommendation. His awards and honors include Doctoral Research Fellowship (Japan Advanced Institute of Science and Technology) and JST SPRING: Support for Pioneering Research Initiated by the Next Generation (Japan Science and Technology Agency).

XIN LIU received the Ph.D. degree in computer science from the Tokyo Institute of Technology. He is currently a Senior Researcher with the Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology, Japan. His research interests include machine learning and data mining for graph data, especially graph and network analysis, graph learning, network science, and web recommendations.

TEERADAJ RACHARAK (Member, IEEE) received the B.Eng. degree (Hons.) from Kasetsart University, Thailand, the M.Eng. degree in computer science from the Asian Institute of Technology, Thailand, the first Ph.D. degree in information science from the Japan Advanced Institute of Science and Technology, and the second Ph.D. degree in engineering and technology from the Sirindhorn International Institute of Technology, Thammasat University, Thailand. He is currently a Senior Lecturer with the School of Information Science, Japan Advanced Institute of Science and Technology. His research interests include trustworthy AI, especially knowledge representation and reasoning, machine learning, neural-symbolic AI, and explainability in AI.

GUANQUN SUN received the first M.S. degree in computer and information science from Hosei University, Tokyo, Japan, in 2019, and the second M.S. degree in software engineering from Tianjin University, Tianjin, China, in 2020. He is currently pursuing the Ph.D. degree in information science with the Japan Advanced Institute of Science and Technology, Ishikawa, Japan. He has been an Assistant Teacher with the Hangzhou Medical College, Hangzhou, China, since 2021. His research interests include medical information engineering, bioinformatics, and medical image processing.

JIANAN CHEN (Graduate Student Member, IEEE) received the B.S. degree in software engineering from the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University (SJTU), in 2017. He is currently pursuing the M.S. degree with the Japan Advanced Institute of Science and Technology. His research interests include low-resource machine translation and knowledge graph embedding.

QIANG MA (Senior Member, IEEE) received the Ph.D. degree from the Department of Social Informatics, Graduate School of Informatics, Kyoto University, in 2004. He was a Research Fellow (DC2) of JSPS, from 2003 to 2004. He joined the National Institute of Information and Communications Technology as a Research Fellow, in 2004. From 2006 to 2007, he was an Assistant Manager with NEC. In October 2007, he joined Kyoto University, where he has been an Associate Professor, since August 2010. His research interests include databases and information retrieval, web information systems, web mining, knowledge discovery, and multimedia mining and search.

LE-MINH NGUYEN received the B.Sc. degree in information technology from the Hanoi University of Science, in 1998, the M.Sc. degree in information technology from Vietnam National University, Hanoi, in 2001, and the Ph.D. degree in information science from the School of Information Science, Japan Advanced Institute of Science and Technology (JAIST), in 2004, where he was an Assistant Professor, from 2008 to 2013. He is currently a Professor with the School of Information Science, JAIST. He leads the Laboratory on Machine Learning and Natural language Understanding, JAIST. He is also the Director of the Intepretable AI Center, JAIST. His research interests include machine learning, natural language understanding, question answering, text summarization, machine translation, legal text processing, and deep learning.