ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230602

Deep Interactions-Boosted Embeddings for Link Prediction on Knowledge Graph

Hong Yin[a], Jiang Zhong[a;*] and Qizhu Dai[a]

[a]College of Computer Science, Chongqing University.
ORCiD ID: Hong Yin https://orcid.org/0000-0002-3854-9686, Qizhu Dai https://orcid.org/0000-0003-1072-7847

Abstract. Link prediction for Knowledge Graphs (KGs) aims to predict missing links between entities. Previous works have utilized Graph Neural Networks (GNNs) to learn specific embeddings of entities and relations. However, these works only consider the linear aggregation of neighbors and do not consider interactions among neighbors, resulting in the neglect of partial indicating information. To address this issue, we propose Deep Interactions-boosted Embeddings (DInBE) which encodes interaction information to enrich the entity representations. To obtain interaction information, we disentangle the representation behind entities to learn diverse disentangled representations for each entity. Then, we learn intra-interactions among neighboring entities in the same component and inter-interactions among different components based on these disentangled representations. With the help of interaction information, our model generates more expressive representations. In addition, we propose a relation-aware scoring mechanism to select useful components based on the given query. Our experiments demonstrate that our proposed model outperforms existing state-of-the-art methods by a large margin in the link prediction task, and this verifies the effectiveness of exploring interactions and adaptive scoring.

1 Introduction

Knowledge graphs (KGs) are collections of large-scale facts represented as structural triples, in the form of (head entity, relation, tail entity), denoted as (u, r, v), which reveal the relations between entities. KGs such as DBpedia [1], Freebase [3], Wikidata [25], and YAGO [20] have been created and widely used in massive intelligence applications, including natural language processing [32], information retrieval [14], and recommender systems [9]. Despite already containing millions of facts, KGs are still incomplete, resulting in poor performance in downstream applications. Therefore, the task of link prediction on knowledge graphs is vital, as it predicts missing facts and further expands existing knowledge graph.

In the field of literature, most state-of-the-art link prediction models are based on knowledge graph embedding. These models aim to learn low-dimensional embedding for entities and relations, and can be broadly classified into three types: translational models, bilinear transformation models, and convolutional network-based models.

Despite their excellent performance, these models have one major limitation - they ignore the meaningful graph topology information. Consequently, Graph Neural Network (GNN) based models have become an active research focus for link prediction. GNNs employ layer-wise propagation to collect neighboring entity's information through a linear aggregation operation. This method improves the expressiveness of the entity representation by utilizing the rich neighboring information. However, such a linear aggregation assumes that the neighboring entities are independent of each other, ignoring the possible interactions between them.

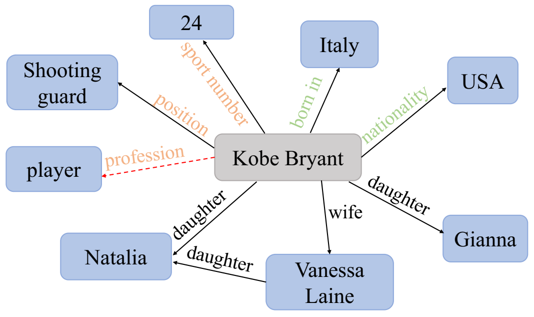

Figure 1. An example of a knowledge graph is shown, where different colors depict a specific semantic aspects. The red dotted line denotes a missing triplet.

Therefore, a straightforward approach is to compute the pairwise interaction information between neighboring nodes and fuse it with aggregated structural information [34]. However, this approach has some limitations when applied to knowledge graphs as it would neglect crucial information — the edge relation, which provides evidence that two entities are connected. In addition, this approach calculates the pairwise interactions between all neighbors, while some interactions may be invalid in a given specific scenario. This arises from the fact that an entity may have multiple aspects of semantics and various entity-relation pairs focus on distinct aspects of entities. In this context, these neighboring entity-relation pairs with the same aspect of semantics form a component. Take the entity *Kobe Bryant* as an example, it has three components: "family", "occupation" and "location", when we have a query *(Kobe, profession, ?)*, we often rely on the interaction information between neighbors under the "occupation" component, i.e., the interaction information between *(position, Shooting guard)* and *(sport number, 24)* and pay less attention to the interaction information between neighbors from other components.

Based on the aforementioned observations, we propose a framework called Deep Interactions-boosted Embeddings (DInBE) for link

[*] Corresponding Author. Email: zhongjiang@cqu.edu.cn

prediction. DInBE is an end-to-end framework in which the encoder learns the representation of entities and the decoder determines the probability of a triplet being true through a relation-aware scoring module. Specifically, our encoder is designed to learn both the structural representation and deep interaction representation of entities. This deep interaction representation consists of intra-interaction and inter-interaction, which are utilized to depict the interaction information between neighbors within the same component and between different components, respectively. To ensure that entity representations under different components are independent of each other, we introduce orthogonality constraints between components. By doing so, the representations are encouraged to be disentangled and better able to characterize the aspect pertinent to a latent component of the entity. Additionally, we utilize a relation-aware scoring mechanism to select the most relevant aspects for the link prediction task, thereby obtaining adaptive scores. By effectively incorporating rich and useful interaction information into structural information, DInBE is able to improve the performance of link prediction.

Our contributions can be summarized as follows:

- We propose a novel framework, called DInBE, that effectively captures both intra-interaction and inter-interaction information to enrich the structural information obtained from the linear aggregation operation.
- We introduce a relation-aware scoring mechanism to better leverage different semantic aspects for the link prediction task.
- Experiments conducted on benchmark datasets show that DInBE outperforms existing methods by a large margin and demonstrate the effectiveness of exploiting interaction information.

2 Related Work

Non-graph Neural Methods for Knowledge Graphs. Non-graph neural methods embed entities and relations into a latent semantic space without utilizing intrinsic topological information. The pioneering model, TransE [4], regards the relation as a translational distance from the head to the tail entity for a triplet. Following this line of research, a series of extended models have been proposed, such as TransH [26] and TransR [13]. In addition, some models operate in a complex space, such as RotatE [21], which exploits the rotation in the complex domain to describe relations. Another line of research measures the plausibility of a target triplet by calculating the semantic similarity in the vector space. The most representative model for this approach is RESCAL [16], which performs a bilinear product on the entity embeddings and relation dense matrix. Based on this schema, a number of studies have been conducted. For instance, DistMult [28] improves RESCAL by converting the dense matrix to a diagonal matrix, and ComplEx [8] further extends DistMult to the complex space. To obtain more deep and expressive representations, convolutional neural networks (CNNs) are introduced due to their strong learning ability and parameter efficiency. Specifically, ConvE [7] applies a convolution over the 2D matrix obtained by reorganizing subjects and relations. HypER [2] simplifies 2D ConvE by utilizing a hypernetwork to generate 1D relation-specific convolutional filters.

Graph Neural Networks for Knowledge Graphs. Although these approaches have been successful, there has been a lack of consideration for structural information in knowledge graphs. Graph neural network-based KG models [17, 18, 29, 5, 15, 24, 30] have attempted to use more graph structure information and develop it into a mature and flexible aggregation pattern that adapts to the characteristics of complex knowledge graphs. R-GCN [17] is the first to apply graph convolution network (GCN) [11] to the LP task, learning entity embeddings under different relations. WGCN [18] introduces learnable relation-specific weights to different neighboring entities. However, the above methods do not consider relation features during the aggregation process. To address this problem, VR-GCN [29], TransGCN [5], KBGAT [15] and CompGCN [24] recursively aggregate the representations of neighboring entities and relations using composition operators. For example, KBGAT concatenates entity and relation embeddings in a triplet to calculate the attention values. Then, the new representation of an entity is obtained by summing every weighted triplet representation. Although structural information is fully utilized by performing linear aggregation, these methods assume that the neighboring entities are independent of each other, with little discussion about the role of interaction information between neighboring entities in characterizing entities.

Table 1. Notations used in the paper.

Notations	Descriptions
$\mathcal{G}$	knowledge graphs
$\mathcal{V}, \mathcal{R}, \mathcal{T}$	set of entities, set of relations, set of triplets
$\widetilde{\mathcal{T}}$	set of negative triplets
x_u	Initial features of entity u
x_u^k	Initial features of entity u in the k-th component
$h_{v,r}^k$	composition features of entity v and relation r in the k-th component
$\mathcal{N}(u)$	set of neighbors of entity u
$\tilde{\mathcal{N}}(u)$	set of u and its neighbors
$\mathcal{R}_{vu}$	set of relations connecting u and v
$\mathcal{T}(u)$	set of neighboring entity-relation pairs of entity u
$h_{u,stru}^k$	the structural representation of entity u in the k-th component
u_{intra}^k	the intra-interaction representation of entity u in the k-th component
u_{inters}^k	the inter-interaction representation of entity u in the k-th component
h_u^k	the final representation of entity u in the k-th component

3 Methods

In this section, we introduce our proposed method in detail. The overall task is to score a triple (u, r, v) in a KG $G = \{\mathcal{V}, \mathcal{R}, \mathcal{T}\}$, that is, to predict the probability that the triple is true, where $\mathcal{V}$ and $\mathcal{R}$ are sets of entities and relations, $\mathcal{T}$ is the set of triplets. An overview of our proposed method is shown in Figure 2. It mainly consists of three parts: 1) structural features learning module to collect the neighboring information through a linear aggregation, 2) interaction features learning module to capture the intra-interaction information and inter-interaction information of entities, 3) relation-aware scoring mechanism to get the score of a target triple being true in a dynamic way. In Table 1, the main notations utilized throughout this paper are summarized.

3.1 Structural Features Learning

This module is designed to facilitate the learning of disentangled representations, in which each component contains distinct semantic aspects. The objective is to generate a representation for a given entity

u that consists of K components, with each component representing a distinct aspect of the entity, such as a person's family. To achieve this, we first project the initialized feature vector x_u into different latent spaces. This enables each component to extract a unique aspect from the entity feature, leading to a disentangled representation.

$$x_u^k = \sigma\left(W_k^T \cdot x_u\right) + b_k. \tag{1}$$

where the initial embeddings $\{x_u^k\}_{k=1}^K$ are obtained from x_u using K distinct projection matrix $W = \{W_1, W_2, \cdots, W_K\}$, σ is the activation function.

Obviously, the information contained is limited and therefore insufficient to achieve disentanglement. To enrich the information, we employ a message passing mechanism and establish an update rule for the k-th component of x_u as follows:

$$h_{u,stru}^k = x_u^k + \text{AGGREGATE}\left(\left\{x_i^k, \forall i \in \mathcal{N}(u)\right\}\right). \tag{2}$$

where AGGREGATE represents the neighborhood aggregation function. $\mathcal{N}(u)$ is the set of neighboring entities.

In this way, $h_{u,stru}^k$ contains information from the k-th aspect of both entity u and all of its neighbors. While common aggregating functions like mean pooling and sum pooling can be used, treating each neighbor equally when determining one component of the representation is not always the most sensible approach. Therefore, an attention mechanism is employed to assign weights to each neighbor, allowing for a more reasonable representation.

In the learning entity representation process, we should not only aggregate neighboring entities as the relations they associated with are of great significance to indicate which semantic aspect the neighbors should belong to. Take the entity "Kobe Bryant" as an example, it appears in the triplets *(Kobe Bryant , position, Shooting guard)* and *(Kobe Bryant , wife, Vanessa Laine)*, where *position* and *wife* can depict the components of "occupation" and "family" respectively. To this end, we propose to compose the neighboring entity and its connected relation, and further learn a weight for them, which indicates the significance of each neighboring entity-relation pair in representing the entity.

In this work, we learn the importance of each neighboring entity-relation pair represented by $b_{u,r,v}^k$ in the k-th component. Specifically, we combine the entity u with its neighboring entity-relation pair and apply a LeakyReLU function to the combination to obtain the absolute attention value of the entity-relation pair. This value indicates the importance of the entity-relation pair and is defined as:

$$b_{u,r,v}^k = \frac{\exp\left(\sigma\left(p \cdot h_{u,r,v}^k\right)\right)}{\sum_{v' \in \tilde{\mathcal{N}}(u)} \sum_{r' \in \mathcal{R}_{v'u}} \exp\left(\sigma\left(p \cdot h_{u,r',v}^k\right)\right)}, \tag{3}$$

$$h_{u,r,v}^k = W\left[h_u^k \| h_{v,r}^k\right], \tag{4}$$

where $\|$ represents the concatenation operation, $h_u^k, h_{v,r}^k \in \mathbb{R}^{\frac{d}{K}}$ are the embeddings of the entity u and composition of entity v and its associated relation r in the k-th component, and d is the embedding size. $W \in \mathbb{R}^{\frac{d}{K} \times \frac{2d}{K}}$ and $p \in \mathbb{R}^{\frac{d}{K}}$ are training parameters. $\tilde{\mathcal{N}}(u)$ represents the extended neighboring entities of entity u and is denoted as $\tilde{\mathcal{N}}(u) = \mathcal{N}(u) \cup u$, and $\mathcal{R}_{v'u}$ denotes the set of relations connecting entities v' and u. σ represents the LeakyReLU function. After the above calculation, we get the attention score $b_{u,r,v}^k$, which represents the weight of the composition of neighboring entity v and its connected relation r when describing the k-th semantic aspect of the entity u.

The aggregated representation of the entity u is the sum of its neighboring entity-relation pairs weighted by their attention values. Now, we formulate the definition of the AGGREGATE function as follows:

$$\text{AGGREGATE} = \sigma\left(\sum_{v' \in \tilde{\mathcal{N}}(u)} \sum_{r' \in \mathcal{R}_{v'u}} b_{v'r'u}^k h_{v'}^k\right). \tag{5}$$

3.2 Interaction Features Learning

Intra-interaction. The semantic of an entity comprises various semantic aspects, each of which represents different properties of the entity. In particular, the intra-interaction information, i.e., the interactions between neighbors, is highly indicative of entity properties. Therefore, we propose a transformer-based approach to capture the interactions of entity u in the k-th component. Specifically, we firstly construct a set $\mathcal{T}(u)$ containing entity-relation pairs. These pairs are formed from the neighboring entities of entity u and their associated relations.

$$\mathcal{T}(u) = \{(r, v) | (u, r, v) \in \mathcal{T}\} \tag{6}$$

And Then, we feed $\mathcal{T}(u)$ into a Nei2seq module, which unpacks these elements within $\mathcal{T}(u)$ and converts it into a sequence $\langle v_1, v_2, \ldots, v_i\rangle$. It should be noted that when converting the set $\mathcal{T}(u)$ to a sequence, we only ensure that the relative location of the relation r and the entity v in the entity-relation pair (r, v) remains unchanged. In contrast, the order of entity-relation pairs is arbitrary since there is no inherent order among neighbors. Afterwards, this sequence is concatenated with the global node [CLS] to explicitly preserve global information, as in [33]. Thus, we obtain the final input sequence $\langle v_{cls}, v_1, v_2, \ldots, v_i\rangle$ of the transformer. Finally, we feed the input sequence into transformer layers to encode the features and represent the [CLS] as the intra-interaction representation of entity u. This approach allows us to obtain node-pair information, such as entity-relation, entity-entity, and relation-relation pairs interactions.

Inter-interaction. Unlike intra-interaction, which focuses on the interaction information within a component, inter-interaction captures the interaction information between components. When an entity u has K different components, it will generate $K-1$ different interaction information under each component. These $K-1$ interaction information pieces contribute differently to the semantic representation of the entity under the current component. To make reasonable use of this $K-1$ interaction information, we employ an attention mechanism to dynamically control the contribution of $K-1$ inter-interaction information under the current component. Specifically, we consider the intra-interaction features and apply a attention mechanism to assess the importance of inter-interaction in learning the inter-interaction feature of entity u in the k-th component.

$$\text{att}_{inters}^{ik} = f^k\left[u_{inter}^{ik} \| u_{intra}^k\right], \tag{7}$$

$$u_{inter}^{ik} = u_{intra}^i \times u_{intra}^k, \tag{8}$$

where $\|$ represents concatenation, f^k is a feedforward neural network that is parameterized by a weight matrix $W_f^k \in \mathbb{R}^{1 \times \frac{2 \times d}{K}}$, followed by a nonlinearity.

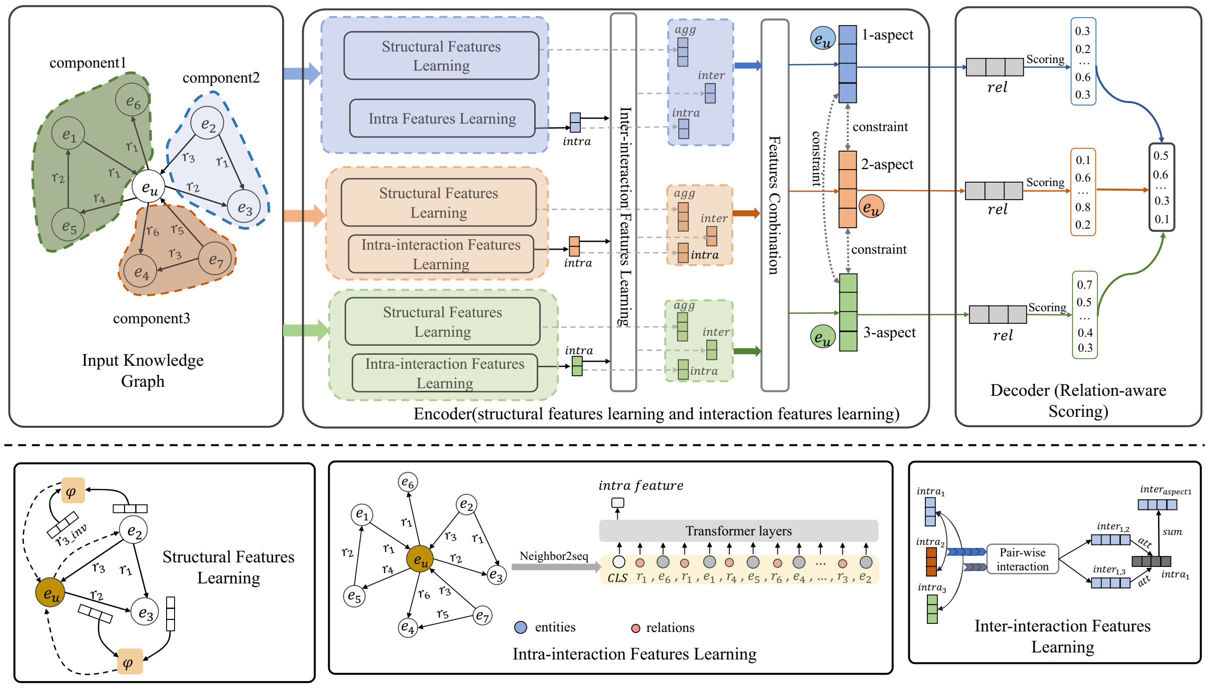

Figure 2. An overview of our proposed DInBE (this example assumes that there are 3 latent components). which consists of three key modules: 1) structural features learning; 2) interaction features learning; 3)relation-aware scoring mechanism. Each color represents a specific component and different components keep individuals with each other by the orthogonality constraint. The final prediction result depends on all components of the entities and is adaptive to the given relation.

We aggregate information from inter-interaction u^{ik}_{inter} to the k-th component of u:

$$u^{k}_{inters} = \sigma_1 \left(\sum_{i \in \{1,2,\cdots,K\} \, and \, i \neq k} \alpha^{ik}_{inters} * u^{ik}_{inter} \right). \quad (9)$$

where σ_1 is a nonlinearity, $*$ denotes multiplication. α^{ik}_{inters} can be seen as contribution of the inter-interaction u^{ik}_{inters} and is generated by applying a softmax function on att^{ik}_{inters}.

3.3 *Relation-aware Scoring Mechanism*

After obtaining the structural features and intra/inter-interaction features of entities under the k-th component, we concatenate them to form the final entity representation h^k_u. The overall process is shown in Algorithm 1. To this end, we adopt the following manner to produce:

$$h^k_u = \left[h^k_{u,stru}; u^k_{inters}; u^k_{intra} \right]. \quad (10)$$

To obtain relation-aware scores, we begin by calculating the score of each candidate triplet under each component. We then derive the final scoring result by considering the attention between the query relation and the component. Specifically, for a given query relation q, a head entity u, and a candidate tail entity v, we compute the score of the triplet under the k-th component as follows:

$$\psi^k_{(u,q,v)} = f(h^k_u; q; h^k_v). \quad (11)$$

where f is a scoring function, such as transE [4], DistMult [28] and ConvE [7]. In this work, we exploit the ConvE as the scoring function, i.e.,

$$\psi^k_{(u,q,v)} = f(\text{vec}(([\overline{h^k_u}; \overline{q}] * \omega))W_s)h^k_v. \quad (12)$$

where $\overline{h^k_u}$ and $\overline{q}$ are 2D reshaped embeddings of h^k_u and q respectively. ω denotes a set of filters of a 2D convolutional layer, $*$ denotes the convolution operator and $\text{vec}(\cdot)$ is a vector concatenation and $W_s \in \mathbb{R}^{\frac{d}{K}}$ is the weight matrix.

After obtaining the score of each triplet under each component, we proceed to adaptively fuse the scores based on the relations. This is because the best-matched component representation should be closer to the given relation embedding, as noted by [27]. To achieve this, we assign a weight to the results obtained from each component.

$$\beta^k_{u,q} = \text{softmax}\left(W_1 h^k_u \cdot W_2 h_q \right) \quad (13)$$

where $\beta^k_{u,r}$ represents the importance of the k-th component of entity u to query relation q. $W_1 \in \mathbb{R}^{d_q \times \frac{d}{K}}$, $W_2 \in \mathbb{R}^{d_q \times d_r}$ are parameter weight matrices. Finally, we can get the final scores for a given query and a candidate tail entity:

$$\psi_{(u,q,v)} = \sum_k \beta^k_{(u,q)} \psi^k_{(u,q,v)} \quad (14)$$

3.4 *Full Objective*

Link prediction. In the training procedure, we leverage a standard cross entropy loss with label smoothing [22]. This is defined as:

$$\mathcal{L}_{lp} = \sum_{(u,r,v) \in \mathcal{T}} -\frac{1}{N} \sum_{i=1}^{N} \left(y_{(u,r,v_i)} \cdot \log\left(\psi_{(u,r,v)}\right) + \left(1 - y_{(u,r,v_i)}\right) \log\left(1 - \psi_{(u,r,v)}\right)\right), \quad (15)$$

in which

$$y(u,r,v) = \begin{cases} 1 & \text{for } (u,r,v) \in \mathcal{T}, \\ 0 & \text{for } (u,r,v) \in \widetilde{\mathcal{T}}, \end{cases}$$

Here $\widetilde{\mathcal{T}}$ is a set of negative triplets created by corrupting the positive triplets set $\mathcal{T}$. The N denotes the number of entities.

Algorithm 1: The proposed encoder, with K components.

```
Input: $x_u \in \mathbb{R}^{d_{in}}$ (the feature vector of entity $u$), and
       $\{x_v \in \mathbb{R}^{d_{in}} : (u,r,v) \in \mathcal{T}\}$ (its neighbors' feature vectors).
Output: $h_u^k \in \mathbb{R}^{d_{out}}$ (entity $u$'s representation under component $k$).
1  for $i \in \{u\} \cup \{v : (u,r,v) \in \mathcal{T}\}$ do
2      for $k = 1, 2, \ldots, K$ do
3          $x_i^k \leftarrow \sigma(W_k^T x_i + b_k)$;
           // Initialize the representations of entity $i$ under each component.
4  for $k = 1, 2, \ldots, K$ do
5      $h_{u,stru}^k \leftarrow x_u^k +$ AGGREGATE
           $(\{x_v^k, \forall v \in \{v : (u,r,v) \in \mathcal{T}\}\})$ // Update.
6  for $k = 1, 2, \ldots, K$ do
7      $\mathcal{T}(u) \leftarrow \{(r,v) | (u,r,v) \in \mathcal{T}\}$
8      $\text{seq}_k \leftarrow \text{Nei2Seq}(\mathcal{T}(u))$, // Convert the neighboring entity-relation pairs into a sequence.
9      $u_{intra}^k \leftarrow \text{transformer}(v_{cls}, \text{seq}_k)$
10 for $j = 1, 2, \ldots, K$ do
11     $u_{inter}^{jk} \leftarrow u_{intra}^j \times u_{intra}^k$,
       $\text{att}_{inters}^{jk} \leftarrow f^k \left[ u_{inter}^{jk} \| u_{intra}^k \right]$, for $k = 1, \ldots, K$;
12     $\left[ \text{att}_{inters}^{j1}, \ldots, \text{att}_{inters}^{jk} \right] \leftarrow$
       $\text{softmax}\left( \left[ \text{att}_{inters}^{j1}, \ldots, \text{att}_{inters}^{jk} \right] \right)$
13 $u_{inters}^k \leftarrow \sigma_1 \left( \sum_{j \in \{1,2,\cdots,K\} \, and \, j \neq k} \alpha_{inters}^{jk} * u_{inter}^{jk} \right)$
14 $h_u^k \leftarrow \left[ h_{u,stru}^k ; u_{inters}^k ; u_{intra}^k \right]$
15 Return $h_u^k$
```

Orthogonality constraint. To enhance the disentangled informativeness as well as to reduce the redundancy between components, we impose an orthogonality constraint to encourage the representations of different aspects to be sufficiently independent. The orthogonality constraint effectively disentangles information of each component and prevents them from contaminating each other. Specifically, let h_u^i and h_u^j be the disentangled representations from the i-th and j-th components, respectively. The orthogonality constraint loss between them is defined as follows:

$$L_{constraint} = \left\| {h_u^i}^\top h_u^j \right\|_F^2, \tag{16}$$

where $\| \cdot \|_F^2$ is the squared Frobenius norm.

Joint training strategy. The final learning objective of our work is defined as the combination of the link prediction loss and orthogonality constraint loss:

$$\mathcal{L} = \mathcal{L}_{lp} + \lambda \mathcal{L}_{constraint},$$

where λ controls the orthogonality constraint. By this joint training strategy, our model is capable of modeling interactions while capturing the adaptive representations of entities.

Table 2. Summary statistics of datasets.

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

Table 3. Details of hyperparameters used for link prediction task.

Hyperparameter	Values
Number of encoding layer	{1,2}
Learning rate	{0.01,0.001,0.005}
Dropout	{0.1,0.2,0.3,0.4,0.5}
Numer of component (FB15k-237)	{1,2,3,4,5,6}
Number of component (WN18RR)	{1,2,3,4}

4 Experiments

4.1 Experimental Configuration

Datasets. We conduct extensive experiments on common link prediction datasets gathered from the literature to evaluate the performance of our proposed method. Specifically, we use WN18RR [7] and FB15k-237 [23], which exclude the inverse relation triplets from the test set to address the issue of data leakage. We use the original data split for all datasets. Table 2 provides a summary of the datasets' statistics.

Evaluation Protocol. To assess the effectiveness of DInBE, we utilize the standard link prediction metrics, as recommended in previous studies. These metrics include Mean Rank (MR), Mean Reciprocal Rank (MRR), and Hits@k. The MR metric computes the average rank value of the test triplets, whereas MRR evaluates the average of the reciprocal of the rank. Hits@k calculates the percentage of all correct answers with a rank lower than or equal to k. A higher MRR and Hits@k score indicates that the rank is more accurately predicted, while a smaller MR suggests that the model is performing well.

Experimental Settings. In our implementation, we select hyperparameters by performing grid search on the validation dataset. Specifically, we set the dimensions of entities and relations as 200 for both the input and output layers, and we find that the optimal number of components K is 4 for FB15k-237 and 2 for WN18RR. During training, we set the following hyperparameters: a batch size of 256, a dropout [19] rate of 0.3, a learning rate of 0.001, and a label smoothing of 0.1, as described in [22].

For all experiments, we initialize all training parameters with Xavier and use Adam [10] as the optimizer to train our model. In addition, we employ an early stopping strategy based on the convergence behavior of the validation set. The model is implemented using PyTorch and runs on a server equipped with an NVIDIA RTX A6000.

Baselines. We compare our model to several state-of-the-art methods, including TransE [4] from translational distance models, DistMult [28] from semantic matching models, CompGCN [24], SACN [18], r-GAT [6], DisenKGAT [27], MRGAT [12] and CorIn [31]

Table 4. Performance comparison of different models on WN18RR and FB15k-237 dataset. We mark the best results with boldness and get the results for all the baseline methods from their previous papers ('-' indicates missing values).

	WN18RR					FB15k-237				
Model	MRR	MR	Hits@10	Hits@3	Hits@1	MRR	MR	Hits@10	Hits@3	Hits@1
TransE	0.226	3384	0.501	-	-	0.294	357	0.465	-	-
DistMult	0.43	5110	0.49	0.44	0.39	0.241	254	0.419	0.263	0.155
SACN	0.47	-	0.54	0.48	0.43	0.35	-	0.54	0.39	0.26
CompGCN	0.479	3533	0.546	0.494	0.443	0.355	197	0.535	0.39	0.264
r-GAT	**0.492**	-	0.578	0.506	0.449	0.368	-	0.558	0.405	**0.276**
DisenKGAT	0.486	**1504**	0.578	0.502	0.441	0.368	179	0.553	0.407	0.275
MRGAT	0.481	-	0.544	0.495	0.449	0.355	-	0.539	0.392	0.266
CorIn	0.479	3321	0.575	0.498	0.447	0.369	170	0.558	0.405	0.275
DInBE(LSTM)	0.49	1830	0.598	0.510	0.458	0.373	186	0.56	0.409	0.275
DInBE(Transformer)	0.491	1783	**0.599**	**0.512**	**0.458**	**0.375**	**163**	**0.561**	**0.411**	0.273

Table 5. Experimental results on prediction by relation type on the FB15k-237 dataset.

		CompGCN		DisenKGAT		DInBE	
Task	Relation Type	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10
Predicting tail entity	1-to-1	0.457	0.604	0.501	0.625	**0.557**	**0.703**
	1-to-N	0.112	0.19	0.128	**0.248**	**0.121**	0.236
	N-to-1	0.471	0.656	0.486	**0.659**	**0.492**	0.653
	N-to-N	0.275	0.474	0.291	0.496	**0.306**	**0.502**
Predicting head entity	1-to-1	0.453	0.589	0.499	0.641	**0.535**	**0.653**
	1-to-N	0.779	0.885	**0.789**	0.889	0.772	**0.889**
	N-to-1	0.076	0.151	0.086	0.18	**0.101**	**0.202**
	N-to-N	0.395	0.616	0.402	0.629	**0.402**	**0.635**

from deep neural network models. TransE and DistMult treat each relation as translational distance and a diagonal matrix, respectively. CompGCN and SACN successfully encode the structural information of entities. r-GAT and DisenKGAT implicitly learn the disentangled representations through encoding KG multiple times. MRGAT and CorIn explicitly separate the neighbors into several components based on relations and learn disentangled representations.

4.2 Main Results

Comparison with baselines. The results of the comparison experiment are presented in Table 4. Our model's performance is superior to all baseline models with respect to most metrics on the WN18RR and FB15k-237 datasets, which demonstrates the effectiveness of our approach in considering interaction information. Neural network models outperform non-neural models on the FB15k-237 and WN18RR datasets due to their ability to capture complex semantic information between entities. Also, our approach outperforms neural network models that solely learn structural information, as it also incorporates interaction information that characterizes entity properties.Furthermore, we observe that methods employing implicit grouping of neighboring entities through learning outperform those that explicitly group neighboring entities. And we also see that the learning of intra-interaction features using LSTM and transformer models results in statistically significant improvements compared to baseline approaches. Interestingly, the transformer approach shows greater gains than the LSTM-based approach, as it excels at extracting interactive features with its impressive capabilities.

In particular, our proposed method DInBE, achieves a significant improvement in MRR, Hits@3, and Hits@10 on the FB15k-237 dataset, which we attribute to the richer interaction information available on complex knowledge graphs. The FB15k-237 dataset has more varied relations and richer neighbors than the WN18RR dataset (19 vs. 2 neighbors per entity on average), making it more representative of real-world scenarios. Therefore, the improvement on FB15k-237 validates the effectiveness of considering interaction information.

Performance by relation category. In this section, we explore the performance of our model under various types of relations, including 1-to-1, 1-to-N, N-to-1, and N-to-N relations. Our study focuses on the FB15k-237 dataset, which offers diverse relation types. Table 5 presents our method's superior performance in predicting both head and tail entities. Notably, all models that learn disentangled representations perform better than the model that only learns structural representations. This indicates that graph neural networks handle complex relations well.

We also observe that predicting tail entities under 1-to-1 and N-to-1 relation types and predicting head entities under 1-to-1 and 1-to-N relation types have higher performance than the other two relation types. This suggests that existing models are capable of capturing simple relations such as combination and inversion effectively. In contrast, the performance of all models in predicting tail entities under 1-to-N relation types and head entities under N-to-1 relation types is the lowest, indicating that existing models struggle to distinguish between more similar entities.

In contrast, our model leverages interaction information and disentangles entity representations, resulting in richer and more distinguishable semantic representations. Therefore, our method outperforms other models in different types of relations.

4.3 Ablation Study

As the proposed method consistently outperforms all kinds of baseline, we investigate the impact of each module of the model to analyze it deeply and comprehensively. To be more specific, we conduct ablation studies on the presence of disentangled features learning (called **DInBE w/o Disentangle**), interaction features learning (called **DInBE w/o Inters**) and orthogonality constraint (called **DInBE w/o Constraint**). Table 6 shows the ablation results for different modules in our models, we can find that all variants of DInBE perform worse than the original DInBE, which demonstrates the effectiveness of each module.

DInBE w/o Disentangle. After removing the disentangled features learning module, the Hits@10 and MRR value reduces. This is because, without disentangled semantic representation, our model becomes limited to calculating the pairwise interaction information of neighboring entities and directly incorporating them into structural information without selection. Such excessively rich interaction information may bring a lot of noise, thus affecting the performance of the model. From Table 6, we notice that removing this module has less impact on the WN18RR dataset as compared to the FB15k-237 dataset. This attributes to the fact that the FB15k-237 dataset includes 237 relation types, making it ideal for tackling issues of multi-relation entanglement. In contrast, WN18RR only has 11 relation types, with each entity representing an extremely narrow meaning.

DInBE w/o Interactions. Based on the results from Table 6, we observe a significant decrease in performance when interaction features are not considered. The analysis of the interaction features learning module reveals that it can capture rich interaction information that plays a crucial role in indicating an entity's characteristics. However, without this interaction information, our model degrades into pure exploitation of structural information, leaving interactions unexplored.

DIN w/o Constraint. To further facilitate the semantic decoupling of entities, we make use of orthogonality constraint to ensure that each component is independent. Intuitively, the absence of separability constraints can easily lead to aggregated information interfering with each other, and thus scenarios are not representative of a given relation. We notice that the removal of the orthogonality constraint results in an average reduction of 0.35% on MRR. This result suggests that considering the independence between components is useful in learning interaction information.

4.4 Impact of component number

In this section, we investigate the impact of varying the number of components on model performance. The appropriate components should be closely related to the real dataset facts. The experimental results for the FB15k-237 and WN18RR datasets are summarized in Figure 3. We observe that when the number of disentangled components is 1, our model essentially becomes a typical model using structural features and intra-interaction features. The interaction information is the combination of all neighbors and has not been filtered, which results in poor performance. Specifically, in the FB15k-237 dataset, increasing the number of disentangled components significantly improves all metrics. The optimal choice is around 4, which is correlated with the "semantic number" of most entities. When $K > 4$, the model's performance dramatically declines. This is likely because the most useful information can be classified into several semantic aspects, and it is challenging to achieve good disentanglement if the number of components is greater than the number of semantic aspects.

In contrast, the WN18RR dataset has a straightforward and finely-grained meaning. As a result, only a small number of components need to be learned. The best selection for K is about 2, and with an increase in the K value, the model's performance drops considerably.

Table 6. Result of ablation study

Model	WN18RR		FB15k-237	
	MRR	Hits@10	MRR	Hits@10
w/o disentangle	0.483	0.587	0.366	0.553
w/o interactions	0.484	0.590	0.369	0.556
w/o constriant	0.487	0.596	0.372	0.56
DInBE	0.491	0.599	0.375	0.561

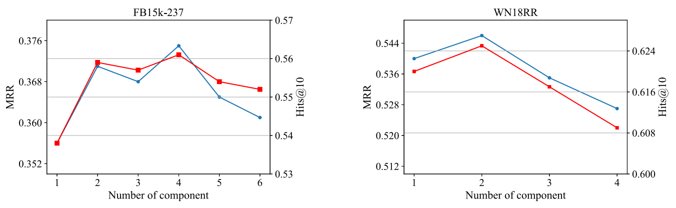

Figure 3. Impact of component number K.

5 Conclusion

In this paper, we propose a novel model called DInBE for link prediction on knowledge graphs, which augments the expressiveness of previous GNN by considering the interactions between neighboring entity-relation pairs based on the disentangled representations. DInBE learns the disentangled representations for each entity and leverages them to learn the intra-interaction information among neighbors in the same component, as well as inter-interaction information among different components, resulting in powerful entity representations. We further propose a relation-aware scoring mechanism to score triplets. The experiments on two benchmark datasets demonstrate our proposed DInBE significantly outperforms several existing state-of-the-art methods for the link prediction task, and verify the effectiveness of capturing interaction information to rich the representations. We also prove the efficiency of different modules through extensive experiments. Future work will consider further exploring semantic information in relations and encoding high-order interactions among neighbors.

Acknowledgements

This work is funded in part by the National Natural Science Foundation of China under Grants No.62176029. This work also is supported in part by the Chongqing Technology Innovation and Application Development Special under Grants CSTB2022TIAD-KPX0206. We express our sincere gratitude to the above funding agencies. We also acknowledge the computational support from the Chongqing Artificial Intelligence Innovation Center.

References

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives, 'Dbpedia: A nucleus for a web of open data', in *The semantic web*, 722–735, Springer, (2007).

[2] Ivana Balažević, Carl Allen, and Timothy M. Hospedales, 'Hypernetwork Knowledge Graph Embeddings', in *Artificial Neural Networks and Machine Learning – ICANN 2019: Workshop and Special Sessions*, eds., Igor V. Tetko, Věra Kůrková, Pavel Karpov, and Fabian Theis, Lecture Notes in Computer Science, pp. 553–565, Cham, (2019). Springer International Publishing.

[3] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor, 'Freebase: A collaboratively created graph database for structuring human knowledge', in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pp. 1247–1250, New York, NY, USA, (2008). Association for Computing Machinery.

[4] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko, 'Translating embeddings for modeling multi-relational data', in *Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2*, pp. 2787–2795, (2013).

[5] Ling Cai, Bo Yan, Gengchen Mai, Krzysztof Janowicz, and Rui Zhu, 'Transgcn: Coupling transformation assumptions with graph convolutional networks for link prediction', in *Proceedings of the 10th international conference on knowledge capture*, pp. 131–138, (2019).

[6] Meiqi Chen, Yuan Zhang, Xiaoyu Kou, Yuntao Li, and Yan Zhang, 'r-gat: Relational graph attention network for multi-relational graphs', *arXiv preprint arXiv:2109.05922*, (2021).

[7] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel, 'Convolutional 2D Knowledge Graph Embeddings', *Proceedings of the AAAI Conference on Artificial Intelligence*, **32**(1), (2018).

[8] Katsuhiko Hayashi and Masashi Shimbo, 'On the Equivalence of Holographic and Complex Embeddings for Link Prediction', in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 554–559, Vancouver, Canada, (2017). Association for Computational Linguistics.

[9] Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li, 'Knowledge graph embedding based question answering', in *Proceedings of the twelfth ACM international conference on web search and data mining*, pp. 105–113, (2019).

[10] Diederik P. Kingma and Jimmy Ba, 'Adam: A Method for Stochastic Optimization', *International Conference on Learning Representations (ICLR)*, (2017).

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, 'Imagenet classification with deep convolutional neural networks', *Communications of the ACM*, **60**(6), 84–90, (2017).

[12] Zhifei Li, Yue Zhao, Yan Zhang, and Zhaoli Zhang, 'Multi-relational graph attention networks for knowledge graph completion', *Knowledge-Based Systems*, **251**, 109262, (2022).

[13] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu, 'Learning entity and relation embeddings for knowledge graph completion', in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 2181–2187, (2015).

[14] Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu, 'Entity-duet neural ranking: Understanding the role of knowledge graph semantics in neural information retrieval', *arXiv preprint arXiv:1805.07591*, (2018).

[15] Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul, 'Learning attention-based embeddings for relation prediction in knowledge graphs', in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4710–4723, (2019).

[16] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel, 'A three-way model for collective learning on multi-relational data', in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pp. 809–816, (2011).

[17] 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*, eds., Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam, Lecture Notes in Computer Science, pp. 593–607, Cham, (2018). Springer International Publishing.

[18] Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou, 'End-to-End Structure-Aware Convolutional Networks for Knowledge Base Completion', *Proceedings of the AAAI Conference on Artificial Intelligence*, **33**(01), 3060–3067, (2019).

[19] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, 'Dropout: A simple way to prevent neural networks from overfitting', *The Journal of Machine Learning Research*, **15**(1), 1929–1958, (2014).

[20] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum, 'Yago: a core of semantic knowledge', in *Proceedings of the 16th international conference on World Wide Web*, pp. 697–706, (2007).

[21] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang, 'Rotate: Knowledge graph embedding by relational rotation in complex space', in *International Conference on Learning Representations*, pp. 801–819, (2019).

[22] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna, 'Rethinking the Inception Architecture for Computer Vision', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, (2016).

[23] 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*, pp. 57–66, (2015).

[24] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar, 'Composition-based Multi-Relational Graph Convolutional Networks', in *International Conference on Learning Representations*, pp. 3061–3076, (2019).

[25] Denny Vrandečić and Markus Krötzsch, 'Wikidata: a free collaborative knowledgebase', *Communications of the ACM*, **57**(10), 78–85, (2014).

[26] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen, 'Knowledge Graph Embedding by Translating on Hyperplanes', *Proceedings of the AAAI Conference on Artificial Intelligence*, **28**(1), (2014).

[27] Junkang Wu, Wentao Shi, Xuezhi Cao, Jiawei Chen, Wenqiang Lei, Fuzheng Zhang, Wei Wu, and Xiangnan He, 'Disenkgat: Knowledge graph embedding with disentangled graph attention network', in *Proceedings of the 30th ACM international conference on information & knowledge management*, pp. 2140–2149, (2021).

[28] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng, 'Embedding Entities and Relations for Learning and Inference in Knowledge Bases', *International Conference on Learning Representations*, (2015).

[29] Rui Ye, Xin Li, Yujie Fang, Hongyu Zang, and Mingzhong Wang, 'A vectorized relational graph convolutional network for multi-relational network alignment.', in *IJCAI*, pp. 4135–4141, (2019).

[30] Hong Yin, Jiang Zhong, Chen Wang, Rongzhen Li, and Xue Li, 'Gs-ingat: An interaction graph attention network with global semantic for knowledge graph completion', *Expert Systems with Applications*, **228**, 120380, (2023).

[31] Mei Yu, Qianyu Zhang, Jian Yu, Mankun Zhao, Xuewei Li, Di Jin, Ming Yang, and Ruiguo Yu, 'Knowledge graph completion using topological correlation and multi-perspective independence', *Knowledge-Based Systems*, **259**, 110031, (2023).

[32] Wenqian Zhang, Shangbin Feng, Zilong Chen, Zhenyu Lei, Jundong Li, and Minnan Luo, 'Kcd: Knowledge walks and textual cues enhanced political perspective detection in news media', in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4129–4140, (2022).

[33] Jianan Zhao, Chaozhuo Li, Qianlong Wen, Yiqi Wang, Yuming Liu, Hao Sun, Xing Xie, and Yanfang Ye, 'Gophormer: Ego-graph transformer for node classification', *arXiv preprint arXiv:2110.13094*, (2021).

[34] Hongmin Zhu, Fuli Feng, Xiangnan He, Xiang Wang, Yan Li, Kai Zheng, and Yongdong Zhang, 'Bilinear graph neural network with neighbor interactions', in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 1452–1458, (2021).