IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 19, 2019, accepted December 28, 2019, date of publication December 31, 2019, date of current version January 9, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2963367

Knowledge Graph Embedding via Graph Attenuated Attention Networks

RUI WANG, BICHENG LI, SHENGWEI HU, WENQIAN DU, AND MIN ZHANG
College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China

Corresponding author: Bicheng Li (lbclm@163.com)

This work was supported in part by the Fujian Social Science Fund Projectunder Grant FJ2017B073, and in part by the Scientific Research Funds of Huaqiao University.

ABSTRACT Knowledge graphs contain a wealth of real-world knowledge that can provide strong support for artificial intelligence applications. Much progress has been made in knowledge graph completion, state-of-the-art models are based on graph convolutional neural networks. These models automatically extract features, in combination with the features of the graph model, to generate feature embeddings with a strong expressive ability. However, these methods assign the same weights on the relation path in the knowledge graph and ignore the rich information presented in neighbor nodes, which result in incomplete mining of triple features. To this end, we propose Graph Attenuated Attention networks(GAATs), a novel representation method, which integrates an attenuated attention mechanism to assign different weight in different relation path and acquire the information from the neighborhoods. As a result, entities and relations can be learned in any neighbors. Our empirical research provides insight into the effectiveness of the attenuated attention-based models, and we show significant improvement compared to the state-of-the-art methods on two benchmark datasets WN18RR and FB15k-237.

INDEX TERMS Knowledge graph embedding, attenuated attention mechanism, neighbor nodes, translational model, graph attention networks.

I. INTRODUCTION

The knowledge graph (KG) is a graph-based data structure composed of ‘‘node-edge-node’’ that represents a semantic network. The node represents a ‘‘concept’’ or ‘‘entity’’, and the edge represents a relation between two entities. For example, in Figure 1, a triple *(Joe Russo, born_in, Cleveland)* is represented as two entities: *Joe Russo* and *Cleveland* with a relation *born_in* linking them. Knowledge graphs are used to describe concepts, entities, and the rich relations between them in the real world. At present, knowledge graphs have been widely used in finance [1], medical [2], semantic search [3], and other fields.

However, the use of knowledge graphs is often limited due to the deficiency in relations arising from incomplete processing data. It is difficult to complete the missing knowledge by means of extraction or fusion because of the sparseness of the data. The completion of true relations, namely knowledge graph completion, remains an active research area. For example, in Figure 1, given triples *(Anthony Russo, brother_of, Joe Russo)*, *(Joe Russo, born_in, Cleveland)*, *(Cleveland, city_of, Ohio)* and *(Ohio, state_of, USA)*, the inference *(Anthony Russo, nationality, USA)* should stands. State-of-the-art methods aim to match entities and relations to low-dimensional continuous vector spaces to characterize their latent semantic features.

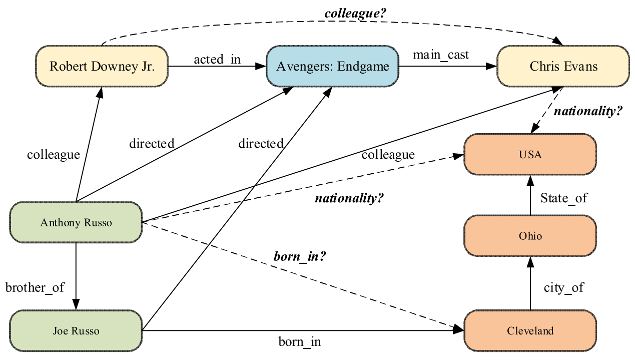


FIGURE 1. A subgraph of a knowledge graph with actual relations between entities(solid lines) and inferred relations that needed to be predicted (dashed lines).

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

The most advanced knowledge embedding approach is knowledge graph representation learning, which mainly categories into tensor decomposition, translational model, and neural networks model. While statistical models need to capture the joint distribution triple features between multiple atomic, which will cause the exponential growth of feature space, the representation learning maps features to the distributed space so that the complex relations are decoupled and dimensionality disaster problem is alleviated. In addition, the data sparsity is prominent in knowledge graph, while the representation learning fills the sparse matrix by numerical calculation, which solves the data sparsity problem to some extent. Finally, representation learning allows symbol data to directly participate in the computation without using statistics such as counts, distributions, and so on.

However, there are still some problems in representation learning. First of all, state-of-the-art methods of representation learning only consider the triples and their hidden features, while the rich information contained in the neighboring triples is not taken into account.

Secondly, the relation embedding is too simple.The existing methods are mainly based on entity embedding, ignoring the influence of the diversity of the relations on the representation of the triple. At last, the existing method adopts a strategy of equally assigning weights to multiple relations on the same path, so that the importance of the relations is treated equally, resulting in error in link prediction.

We use event knowledge graph to make prediction of events, so the accuracy of relations is critically significant. Once the relation reasoning on the critical path goes wrong, it may lead to the prediction of completely oppsite information, misleading the decision analysis. Therefore, improving the accuracy of relation prediction is an urgent problem we need to solve. Inspired by the previous research, we propose the attenuated attention-based graph embedding for knowledge graph completion. Graph attention networks(GATs) with n-hop neighbors [14] have achieved improvements in relation prediction. Our model introduces the attenuated attention mechanism while considering the n-hop neighbors, i.e., the closer the entity is to a given entity, the higher the weight of attention gained. We add it to the GATs and learn the new embedding of entities and relations. Then we use our model to train the relation and entity embeddings separately. More details will be described in Section III.

Our contributions are as follows. First, in order to learn the more expressive embedding, we introduce the graph attention network. Secondly, in view of its existing limitations, we propose the attenuated attention mechanism, and then propose a novel entity and relation embedding method based on graph attenuated attention networks. Third, we use the information of the n-hop neighbor nodes to extend the representation of the entities and relations. Fourth, we introduce the encoder-decoder model. The graph attenuated attention network is used as the encoder, and the Capsule Networks Embedding(CapsE) [18] is used as the decoder. This model supports the exploration of the triple features on a deeper level. Finally, we evaluate the proposed approach with the experiment that uses two benchmark datasets. The experimental results show that our model accuracy performs better than the state-of-the-art methods on the FB15k-237, and most indicators in WN18RR are better than the state-of-the-art methods.

The rest of the paper is structured as follows. We first review the related work in Section II and then present our detailed approach in Section III. The experimental dataset descriptions, results, and analysis are reported in Section IV and followed by our conclusion and future work in Section V.

II. RELATED WORK

Recently, several different representation learning methods have been proposed for relation prediction. These methods can be broadly classified into (i) tensor decomposition, (ii) knowledge embedding models.

The basic idea of tensor decomposition is to replace the original relation matrix with multiple low-dimensional matrices or tensor products, thus replacing sparse and large amounts of raw data with a small number of parameters. The RESCAL [6] takes the inherent structure of dyadic relational data into account by employing the tensor factorization. The Neural Tensor Networks (NTN) [4] model expresses the relations as a matrix to characterize the correlation of potential features. The bilinear matching between the entities (head, tail) and the relations are used to judge the possibility of the relation being established. However, both of these models require a large number of matrix multiplication operations, which greatly increases the time cost.

The knowledge embedding models are further classified as translational models and neural network models. The translational models are based on the energy function, which demostrate the fact that golden triples have low energy while corrupted triples have high energy [8]. The judgment of the established tirple is golden or not is determined by the calculation of the energy function. Translating Embedding(TransE) [5] is a simple model with fewer parameters, but there are certain problems in dealing with one-to-many and many-to-one relations. In order to solve the problem, Translating embedding on Hyperplanes(TransH) [7] and Translating Embedding in Relation spaces (TransR) [8] models are proposed. TransH and TransR calculate a triple score with the same entity using alternative representations in different relation spaces, effectively avoiding the convergence problem. Translating Embedding via Dynamic Mapping Matrix(TransD) [9] solves the diversity of entities and relations by applying the transitional matrix determined by the corresponding entities and relations. Since the initial introduction of TransE in 2013, a variety of methods have been proposed under this framework. The community has also proposed additional methods (e.g., Translating Embedding via relational Mapping properties(TransM) [12], Translating Embedding via Adaptive Approach(TransA) [10]). A summary is shown in Table 1. The scoring functions of the methods are introduced, and the number of parameters represents the model complexity.

TABLE 1. Scoring function and complexity of proposed translational embedding models. n_e and n_r represent the number of entities and relations, respectively. d is the dimension of entity and relation embedding. h, r, t represent the head entity, relation, and tail entity embedding, respectively, M_r represents the relation transitional matrices, I denotes the identity matrix.

Model	Scoring Function	Parameters
TransE[5]	$\|\| h+r-t \|\|$	$d(n_e+n_r)$
TransH[7]	$\|\| (h-M_r^T hM_r)+r-(h-M_r^T hM_r) \|\|$	$d_e n_e + d_r n_r$
TransR[8]	$\|\| M_r h + r - M_r t \|\|$	$d_e n_e + (d_r + d_r)_r^n$
TransD[9]	$\|\| (h^{'T} M_r + I)h + r - (t^{'T} M_r + I)t \|\|$	$2d_e n_e + 2d_r n_r$
TransM[12]	$M_r \|\| h+r-t \|\|$	$d(n_e+n_r)+n_r$
TransA[10]	$(h+r-t)^T M_r (h+r-t)$	$d_e n_e + (d_r + d_r)_r^n$

Different from the translational models, in the field of neural networks, the representation of the knowledge graph is how to express entities and relations through neural network models so that entities and relations can be calculated by symbols. The neural networks models, which focus more on mining latent semantic information of triples, mainly include Neural Tensor Networks (NTN) [4], Convolutional Knowledge Base Embedding(ConvKB) [16], Convolutional Network Embedding(ConvE) [15], Relational Graph Convolutional Networks (R-GCN) [21], Graph Convolutional Networks(GCNs) [11], and Capsule Networks Embedding (CapsE) [18]. ConvKB uses the internal structure of the textual relation as input to the convolutional neural network. NTN learns a tensor network for each relation in the knowledge graph. ConvE uses convolutional neural networks (CNNs) to optimize the input vector to score the triples. The R-GCN uses a graph convolutional network to obtain an embedding of the triples, then applies DistMult [19] to compute a score for the embeddings. In the European space represented by the image, the number of neighbors of the node is fixed. However, in the non-European space represented by the knowledge graph, neighbors are not fixed. The convolution operation in the European space is a feature of the pixel extracted by a fixed-size learnable convolution kernel. Nevertheless, the traditional convolution kernel cannot be directly used to extract the features of the nodes on the graph due to the unfixed number of neighbors. Therefore, GCNs was introduced to find a learnable convolution kernel suitable for graphs. CapsE represents each triple as a 3-column matrix and feeds into capsule neural networks to generalize the expressive embedding. A key limitation of tensor decomposition and neural network based approaches is the high computational cost. To address this limitation, the holographic embedding model(HOLE) [17] has been proposed to construct a more efficient embedding representation by applying the cyclic correlation of entity embedding. Besides, taking path learning into consideration, other than [14], reinforcement learning(RL) with multi-hop knowledge embedding is proposed for the use of query answering based on knowledge graph [28]. Relation path embedding(RPE) [31] adds multi-hop relation paths to the translation model and simultaneously embeds each entity into two types of latent spaces. Path-based Attribute-aware Representation Learning model (PARL) [32] is proposed to perform path denoising and path representation learning for the relation prediction task. Discriminative path-based embedding model (DPTransE) [33] builds interactions from the jointly learned latent features and graph features and uses the graph features as the crucial prior to offer precise and discriminative embedding. Meta-based multi-hop knowledge graph reasoning (Meta-KGR) [34] adopts meta-learning to learn effective meta parameters from high-frequency relations that could quickly adapt to few-shot relations.

In addition to the models above, the incorporation between attention mechanisms and neural networks, which aim to improve the robustness of models, is widely studied. Attention Graph Convolution Network (AGCN) [35] consists of an attention mechanism layer and Graph Convolution Networks(GCNs) to perform superpixel-wise segmentation in big SAR imagery data. Graph Attention Model (GAM) [36] focuses on small but informative parts of the graph, avoiding noise in the rest of the graph. Attention mechanism is applied to reason evidence from the representation of multiple paths to predict whether the entities should be connected by the candidate relation [41]. Besides, attention mechanism is also applied to the classic compositional method to find reasoning paths between entities [40]. Open-domain conversation generation model with graph attention model [37] retrieves relevant knowledge graphs from a knowledge base and then encodes the graphs with a static graph attention mechanism. Learning node embeddings via graph attention utilizes attention parameters exclusively on the data itself instead of inference. Knowledge aware Path Recurrent Network (KPRN) [38] generates path representations by composing the semantics of both entities and relations and allows effective reasoning on paths to infer the underlying rationale of a user-item interaction for recommendation. Knowledge Graph Attention Network (KGAT) [39] recursively propagates the embeddings from a node neighbors to refine the node embedding, and employs an attention mechanism to discriminate the importance of the neighbors for recommendation as well.

In summary, although the neural network-based methods suffer the problems of high complexity and computational cost, the triples based on them have a stronger expressive ability. But state-of-the-art models only regard the relation embedding as an auxiliary feature of the entity embedding, and does not deeply consider how to embed the relations. Besides, when there are indirect connections (not 1-hop relation) between the two entities, the former research assigns the same weight to the relations of each hop, resulting in the loss of information about the partial neighbor triples. At this time, a model that integrates the neural network model and translational model, leveraging the advantages of both models, is not yet available in the literature. Here, we propose a combined model to further improve the task of knowledge graph completion in terms of prediction accuracy and computational cost.

III. APPROACH

Here, we first formalize the notation adopted in this work. A knowledge graph is represented by $G = (E, R)$, where $E = \{e_1, \ldots, e_i, \ldots, e_n\}$ and $R = \{r_1, \ldots, r_i, \ldots, r_m\}$ denote the set of entities and relations, respectively. The triples are represented as (e_i, r_k, e_j) where e_i denotes a head entity, e_j denotes a tail entity, and r_k denotes a relation linking e_i and e_j. Their embeddings are denoted by $(\boldsymbol{h_i}, \boldsymbol{r_k}, \boldsymbol{t_j})$. The embedding representation model attempts to learn efficient representations of the entities, relations and scoring functions f, such that for a given triple $a = (e_i, r_k, e_j)$, $f(a)$ gives the probability of a being a golden triple.

A. GRAPH ATTENTION NETWORKS

As we introduced in Section II, GCNs make a great progress in graph features extraction. But there remains a problem that GCNs only focus on the node itself without considering its neighbors, which contain rich valuable information. The graph attention network (GATs) [13] is a further improvement of GCNs. The GATs emphasize the assignment of varying importance values to different node neighbors, rather than allocating an equal weight for all neighbor nodes, as is done in GCNs.

The input feature of the node-set in the graph attention network layer is $e = \{e_1, e_2, \ldots, e_N\}$, the output features of the nodes are $e' = \{e'_1, e'_2, \ldots, e'_N\}$, where e_i indicates the input embedding of the i-th node. e'_i indicates the output embedding of the i-th node. N indicates the number of nodes. The attention value of a node can be formalized as

$$e_{ij} = f_a(We_i, We_j) \tag{1}$$

where W represents a parametric linear transformation matrix that maps input features to high-dimensional output feature space. The attention value e_{ij} indicates the edge between node e_i and node e_j and f_a represents an attention function.

The attention value reflects the importance of the edge (e_i, e_j), which can be used to measure the importance of the head node e_i. The attention value can be obtained as it shown in Figure 2. We learn a weight of attention for each edge and then gather information from neighbors based on those

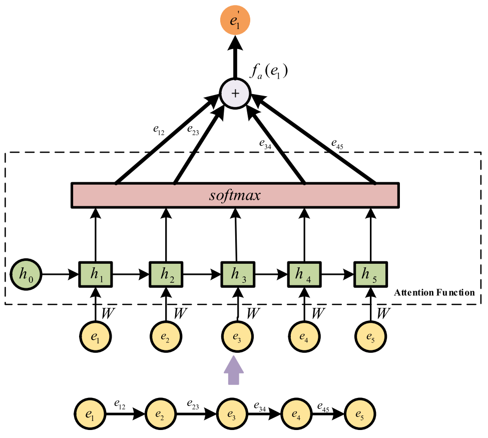


FIGURE 2. An example to explain how GATs works. The entity e_1 gathers information from its neighbors and use attention function to assign different weights to different relations.

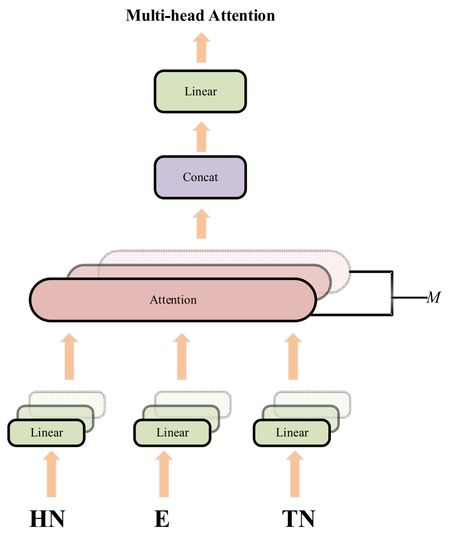


FIGURE 3. Multi-Head Attention consists of several attention layers running in parallel.

weights as

$$e'_i = \sigma\Big(\sum_{j \in \Omega_i} \eta_{ij} We_j\Big) \tag{2}$$

where, η_{ij} represents relative attention, which computed by a softmax function over all the values in the neighbors. W represents a linear mapping matrix. After this operation, the neighbors representation of the node e_i is output. The η_{ij} can be calculated as follows

$$\eta_{ij} = softmax_j(e_{ij}) = \frac{exp(e_{ij})}{\sum\limits_{n \in \Omega_i} \sum\limits_{r \in \Re_{in}} exp(e_{in})} \tag{3}$$

where, Ω_i denotes the neighbors set of entity e_i, $\Re_{in}$ denotes the relations set which connects between e_i and e_n.

To prevent over-fitting of the model, we use multiple independent different attentions for the attention calculations [22]. The multi-head attention structure is shown in Figure 3.

The head node (HN), edge (E), and tail node(TN) first enter a linear transformation and then are input to the expansion point product attention for M times, namely the multi-head attention. For each of the M times, the parameters between the headers are not shared, and the parameters for the linear transformation of HN, E, and TN are different. Then, the M-th order expansion results are concatenated, and the values obtained by the linear transformation are used as the result of the multi-head attention. The multi-head attention process of concatenating M attention heads is shown as follows

$$e_i' = \overset{M}{\underset{m=1}{||}} \sigma(\sum_{j \in \Omega_i} \eta_{ij}^m W^m e_j) \tag{4}$$

where, $||$ represents a concatenate operation, σ represents a nonlinear activation function, η_{ij}^m represents the weight obtained by the m-th attention mechanism, and W^m represents a linear mapping matrix of the m-th attention mechanism. Then, we use the averaging method to obtain the final node representation as follows

$$e_i' = \sigma(\frac{1}{M} \sum_{m=1}^{M} \sum_{j \in \Omega_i} \eta_{ij}^m W^m e_j) \tag{5}$$

B. ATTENUATED ATTENTION MECHANISM

The attention mechanism involved in deep learning essentially plays a similar role as the human selective visual attention mechanism [28]. The core is to select critical features for the current mission objectives from sufficient information. However, from the perspective of human experience, the scope of influence of attention weight is not equivalent.

For example, people always pay more attention to objects that are close to themselves, attention to objects that are further away decrease. As shown in Figure 3, *Angelina Jolie*'s 4-hop neighbor *Canada* should have a much smaller impact than the 1-hop neighbor *Brad Pitt*. Therefore, in a knowledge graph, the closer an entity is to a given entity, the higher its attention value obtains. Besides, Figure 4 also demonstrates that the lighter the color of the relation in the graph, the lower the weight and the smaller the impact on a given entity. Based on this, we propose the attenuated attention mechanism to assign different weights to the n-hop neighbors attention values of a given entity.

Following the principle that the closer to a given entity, the higher weight of attention is gained, we define the attenuated attention coefficient θ_{id}, which denotes the attenuation of the d-th hop neighbor for a given node e_i. Suppose the distance between a given node and their d-hop neighbor is x_d, so the attenuated attention coefficient can be formalized as follows

$$\theta_{id} = \theta_0 exp(-\frac{x_0}{x_d}) \tag{6}$$

where, θ_0 denotes the initial attenuated attention coefficient, and x_0 represents the path step length, default is 1.

We incorporate the attenuated attention mechanism into GATs to form our proposed Graph Attenuated Attention Networks (GAATs) model. The model gathers features from the neighbors of nodes by assigning different weights to different relations to reinforce the entities and relations embedding.

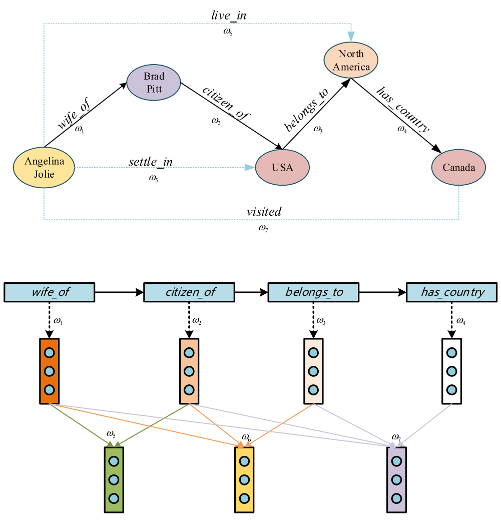


FIGURE 4. The figure shows the aggregation process of our graph attenuated attention layer. w_i denotes relative attention values of an edge. The blue dashed lines represent an auxiliary edge from n-hop neighbors. The attention weight of node "*Angelina Jolie*" is $\omega_1 + 1/2\omega_2 + 1/3\omega_3 + 1/4\omega_4$, in case n = 4.

C. RELATION EMBEDDING WITH GAATs

State-of-the-art methods are focused on entity embedding, while relation embedding only using TransE-trained initialization vectors. However, as the most important part of the knowledge graph, relation plays a decisive role in the quality of knowledge reasoning. In addition, specifical relation embedding can make representation embedding containing more valuable features. Therefore, we propose to refine the type of relations and use our GAATs model to learn the embedding of relations.

We define the initial relations embeddings matrix $R \in \mathbb{R}^{N_r \times T}$, where N_r represents the number of relations, T represents the initial feature dimension embedded in each relation, and the i-th row represents the embedding of the i-th relation. The corresponding output matrix after the GAATs layer processing is $R' \in \mathbb{R}^{N_t \times T'}$, where N_t represents the number of triples, and T' represents the output feature dimension. In other words, we learn relations embeddings independently for each triple.

We use the GAATs model to learn the relation embedding. In the graph structure of the relations, we first calculate the attention value of each relation separately, and obtain the output relation embedding matrix as $R' = R \cdot W^R$, where $W^R \in \mathbb{R}^{T \times T'}$ denotes the weight matrix. Specifically, extracting some of the subgraphs in Figure 1, we divide the relations into three categories as it shown in Figure 5.

(1) If there is a direct relation between the two entities, as shown in Figure 5(a). Then, the relation is expressed as

$$R_{ij}' = \eta_{ij} R(cor) \tag{7}$$

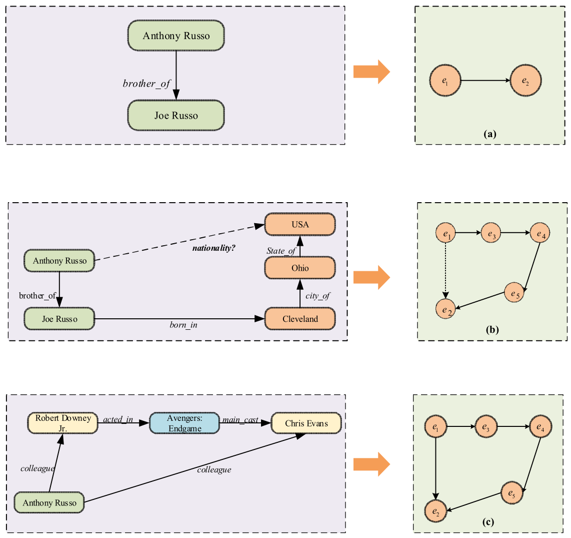


FIGURE 5. Three cases for the GAATs model. (a) direct relation between entities; (b) indirect relation without direct connection; 4(c) indirect relation with direct connection.

where, R'_{ij} indicates the relation vector between the entities e_i and e_j, η_{ij} denotes attention value between the entities e_i and e_j, and $R(cor)$ represents the corresponding row of the initial relation matrix.

(2) If there is an indirect connection between the two entities, but they are accessible through the neighbor nodes, as shown in Figure 5(b). Then, the relation is expressed as

$$R'_{ij} = \sum_{s \in \Omega_i} \theta_{is} \eta_{is} R(cor) \tag{8}$$

where, Ω_i denotes the neighbors set of entity e_i, θ_{is} denotes the attenuated attention coefficient between the entities e_i and e_s. We use the attenuated attention mechanism to calculate the neighbor weighted attention values and then use the product between the relation embeddings of the neighbors and the attention values as the representation of the relation.

(3) If there is an indirect connection between the two entities, and a direct connection exists, as shown in Figure 5(c). We use the attenuated attention mechanism to calculate the neighbor attention values and then use the relation of the existing edge for its representation. Then, the relation is expressed as follows

$$R'_{ij} = (\sum_{s \in \Omega_i} \theta_{is} \eta_{is}) \cdot R(cor) \tag{9}$$

However, while learning the new relation embedding, the relations lose their initialized information. To address this issue, we assign two matrices in order to add the original relation embedding matrix into the final representation as

$$R'' = W^N R W^T + R' \tag{10}$$

where, $W^N \in \mathbb{R}^{N_t \times N_r}$ and $W^T \in \mathbb{R}^{T \times T'}$ denote linear transformation matrices, and R'' represents the final relation embedding matrix.

D. ENTITY EMBEDDING WITH GAATs

Entities play an important role in knowledge graphs, and the correspondence between entities and relations is especially important in knowledge graph reasoning. For example, the entity *Anthoy Russo* plays different role in different triples, i.e. *(brother_of, Joe Russo)*, *(college, Robert Downey Jr.)*. Entity Embedding with GAATs has the following advantages: (i) Our model can solve the existence of insufficient flexibility of the translational models. (ii) Our model can embed the same entity under different relations. (iii) Our model calculates the attention weight of an entity using the attenuated attention mechanism, which allows entity carry the distance information of the relation path in the knowledge graph, to concrete entity embedding and easier to handle complex relations such as 1-N, N-1 and N-M.

We define the initial entity embedding matrix $E \in \mathbb{R}^{N_e \times D}$, where N_e represents the number of entities, D represents the initial feature dimension of the entity embedding, and the j-th row represents the embedding of the j-th entity. The corresponding output matrix after the GAATs layer processing is $E' \in \mathbb{R}^{N_e \times D'}$, where D' represents the output feature dimension.

The attention value reflects the importance of the edge feature [20], which can be used to measure the importance of the head entity. We first learn the embedding of the triple in KGs. We concatenate the initialized entity embedding with the relation vector of Section III.C to obtain the initial representation t_{ijk} of the triple, as follows

$$t_{ijk} = W'[h_i||t_j||r_k] \tag{11}$$

where, $W' \in \mathbb{R}^{1 \times (2D+T')}$ denotes linear transitional matrix, t_{ijk} indicates the vector of triple (e_i, r_k, e_j), and $\boldsymbol{h_i}, \boldsymbol{t_j}, \boldsymbol{r_k}$ represent the embedding of e_i, r_k, e_j, respectively.

Then, we use the *ReLU* activation function to learn a triple embedding to ensure that the triple attention is non-negative.

$$c_{ijk} = ReLU(Vt_{ijk}) \tag{12}$$

where, c_{ijk} means vector after transformation and $V \in \mathbb{R}^{1 \times (2D+T')}$ denotes the linear transformation parameter matrix.

The softmax function is applied to obtain the attention values of the entity's n-hop neighbors, thus calculating the attention weight η_{ijk} of each triple. Meanwhile, as we learn the weight of each triple attention, we add our attenuated attention coefficient to ensure that the closer we get to the given triple, the higher the weight gains. η_{ijk} can be calculated as follows

$$\eta_{ijk} = softmax_{jk}(\theta_{ijk} c_{ijk}) = \frac{exp(\theta_{ijk} c_{ijk})}{\sum_{n \in \Omega_i} \sum_{r \in \Re_{in}} exp(\theta_{inr} c_{inr})} \tag{13}$$

where, Ω_i denotes the neighbors set of entity e_i, $\Re_{in}$ denotes the relations set which connects between e_i and e_n, and θ_{ijk} is the attenuated attention coefficient of the triple (e_i, r_k, e_j). The new embedding of the entity e_i is updated to the weighted sum of the product between the triple attention weight and the triple embedding, as follows

$$h_i' = \sigma(\sum_{j\in\Omega_i}\sum_{k\in\Re_{ij}} \eta_{ijk} t_{ijk}) \tag{14}$$

In order to improve the stability of the learning process and learn more neighbor information, we add a multi-head attention mechanism similar to GATs. Finally, the neighbors of each entity are represented as follows

$$h_i' = \sigma(\frac{1}{M}\sum_{m=1}^{M}\sum_{j\in\Omega_i}\sum_{k\in\Re_{ij}} \eta_{ijk}^m t_{ijk}^m) \tag{15}$$

Similar to relation embedding, while learning the new entity embedding, the entities lose their initialized information. To address this shortcoming, we assign a linear transitional matrix $W^E \in \mathbb{R}^{T\times T'}$ in order to add the original relation embedding matrix into the final representation as follows

$$E'' = W^E E + E' \tag{16}$$

This is the graph attenuated attention layer shown in Figure 6. In our structure, the entity gathers the neighbors' information layer by layer. For example, in Figure 4, the entity *Brad Pitt* gathers information from the direct neighbor in the first layer. Then, it gathers information from the indirect entity *USA* and the direct entity *Angelina Joile* in the second layer. In general, an n-hop-neighbor entity can be accumulated by an n-layer model.

E. TRAINING OBJECTIVE

Our model draws on the scoring function of the translational model, in which the basic idea is the condition $h_i + r_k \approx t_j$ holds for a golden triple (e_i, r_k, e_j). Specifically, we try to learn the entities embeddings and relations embeddings by minimizing the L1-norm dissimilarity measure given by $d(h+r,t) = ||h_i + r_k - t_j||_1$.

We train our model with the margin-based ranking loss as the objective for training as follows

$$L = \sum_{(h,r,t)\in T}\sum_{(h',r,t'\in T')} [\gamma + d(h+r,t) - d(h'+r,t')]_+ \tag{17}$$

where, $[x]_+ \triangleq \max\{x,0\}\gamma > 0$ is a hyper-parameter, T denotes the set of golden triples, and T' denotes the set of corrupted triples, given formally as

$$T' = \{(e_{i'}, r_k, e_j)|e_{i'} \in \varepsilon\backslash e_i\} \cup \{(e_i, r_k, e_{j'})|e_{j'} \in \varepsilon\backslash e_j\} \tag{18}$$

F. DECODER

In order to extract the latent features inside the triples and analyze the global embedding properties of a triple across each dimension, we apply CapsE [18] as a decoder.

Similar to ConvKB, CapsE first learns a three-column matrix for the triples. Each column represents the vector of the head entity, relation, and tail entity. Then, the convolution operation is performed on the matrix using multiple convolution kernels to extract the deep relations within the triples. The difference with ConvKB is that the capsule networks are added behind the feature vector. After the feature vector transformation, it becomes the first layer of capsule network, and each vector is a capsule. After completing the routing process and the activation function, the information is passed to the capsule network of the second layer. There is only one capsule in the second layer, and the modulus of the vector corresponding to the capsule is the probability that the (e_i, r_k, e_j) triple exists in the knowledge graph. The score function can be written formally as

$$f(e_i, r_k, e_j) = ||capsnet(ReLU([e_i, r_k, e_j] * \Omega))|| \tag{19}$$

where, *capsnet* represents the capsule network operation, Ω represents the convolutional layer filter set of the shared parameters, and * represents the convolution operation. The model is trained using the Adam optimizer and the soft margin loss as follows

$$L = \sum_{(e_i,r_k,e_j)\in(T\cup T')} \log(1+\exp(-t_{(e_i,r_k,e_j)} \cdot f(e_i, r_k, e_j))) \tag{20}$$

where,

$$t_{(e_i,r_k,e_j)} = \begin{cases} 1 & for\ (e_i, r_k, e_j) \in T \\ -1 & for\ (e_i, r_k, e_j) \in T' \end{cases}$$

IV. EXPERIMENTS RESULTS AND ANALYSIS

A. DATASETS

To evaluate our model, we used two benchmark datasets: WN18RR [15] and FB15k-237 [15]. The knowledge graph completion task mainly consists of link prediction and triple classification. In addition, in order to testify the performance of our model, our apply a task named relation prediction to make sure our model can handle the inference of relations. The research shows that when the link prediction task is performed on the datasets WN18 and FB15k, it is affected by the inverse relation, which is a simple inverse-rule-based model can be used to obtain optimal results. Therefore, the corresponding sub-datasets WN18RR and FB15k-237 are used to solve the problem of the reversible relation between the datasets [27]. We use the dataset partitioning method to divide the training set and the test set. Table 2 provides all the information we use for the datasets.

B. EVALUATION PROTOCOL

The goal of the link prediction task is to predict a golden triple when the head or tail entity is missing, i.e., given (r_k, e_j) to predict e_i or given (e_i, r_k) to predict e_j. In this task, we remove the head or the tail entity and replace it with all other entities in the corpus. We first calculate the scores of these corrupted triples and rank them in descending order. The ranking of the correct entity is finally recorded. The link prediction emphasizes the final ranking of the correct entity rather than just finding the best entity. Similar to TransE, we also use two methods as our evaluation strategy: MeanRank (MR) and

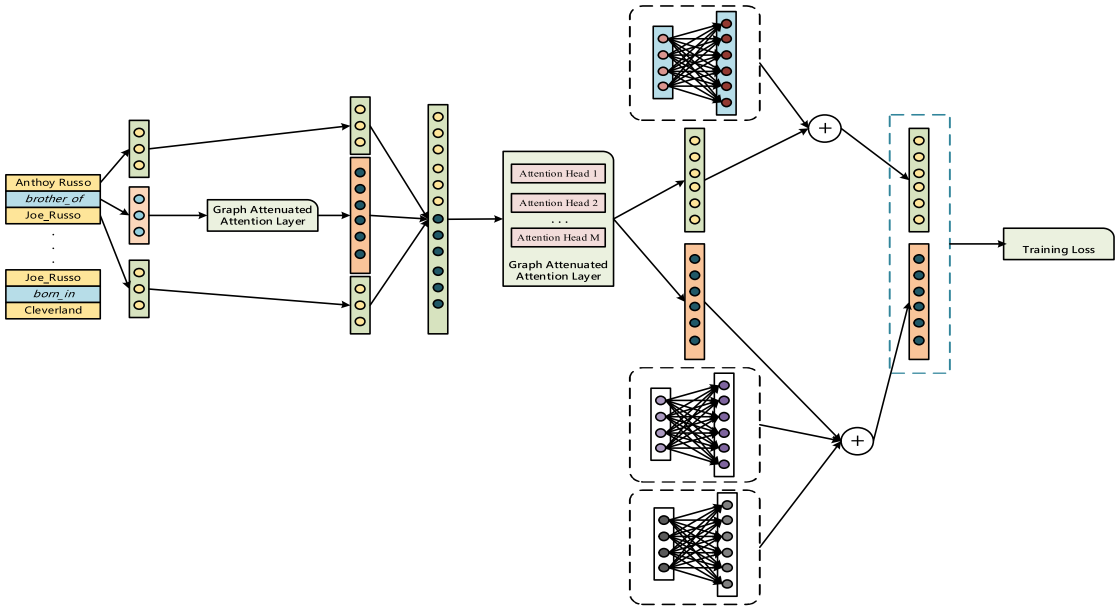


FIGURE 6. This figure shows the whole structure of our model. Yellow circles denote initial entity embeddings, blue circles mean initial relation embeddings, dark yellow and blue circles represent transformed entities and relation embeddings, respectively.

TABLE 2. Dataset description.

dataset	#entity	#relation	#edge			mean in-degree	Median in-degree
			Train	Valid	Test		
WN18RR	40,943	11	86,835	3034	3134	2.12	1
FB15k-237	14,541	237	272,115	17,535	20,466	18.71	8

the proportion of the top N correct entities (Hit@N), N $= 1, 3, 10$. In addition, we also use the Mean Reciprocal Rank (MRR) commonly used in information retrieval as another strategy for evaluation. The lower MR values and the higher MRR or Hit@N values demonstrate better accuracy performance of the model [23]. We call this evaluation method "Raw". Note that there may also be corrupted triples in the KG, and these corrupted triples will be considered as the correct triples. Therefore, before ranking, we should delete the corrupted triples contained in the training set, the validation set, and the test set. This evaluation method is called "Filter". In this article, we summarize the evaluation results of using the two methods, "Raw" and "Filter".

C. TRAINING PROTOCOL

We create two sets of corrupted triples using the method of replacing the head or tail entity of a golden triple. One of them is a collection that replaces only the header entities, and the other is a collection that replaces only the tail entities [26]. To ensure the robustness of the link prediction, we use the classical Bernoulli sampling method, and the two sets extract an equal number of corrupted triples as training samples.

We use the TransE method to initialize the entity embedding matrix and the relation embedding matrix. First, we use the GAATs model to train the embedding of entities and relations in the datasets., and then we train a decoder CapsE to complete our link prediction task. The attenuated attention mechanism is introduced while joining the n-hop neighbor. We use the Adam optimizer to determine the values for all of the parameters. Among them, the initial learning rate is set to $\lambda = \{0.001, 0.01, 0.1\}$, the embedding dimension of the entity and relation is set to $k = \{50, 100, 150, 200\}$, and the margin is set to $\gamma = \{1, 2, 10\}$. The optimal parameters are configured as: $\lambda = 0.01$, $k = 100$, $\gamma = 2$ on the WN18RR dataset; $\lambda = 0.01$, $k = 100$, $\gamma = 1$ on the FB15k-237 dataset.

D. RESULT AND ANALYSIS

We have selected eight current, state-of-the-art methods: DisMult, ConvE, CapsE, TransE, TransH, ConvKB, R-GCN, and n-hop-GATs. Table 3 and Table 4 compares the results of our model experiments with the state-of-the-art results published previously using the same evaluation protocol. The experimental results show that our model performs better than the state-of-the-art method in Hit@N on WN18RR; the performance is better than the state-of-the-art method on FB15k-237. In particular, our Hit@N has increased by 3% overall on the FB15k-237. The results also show that all of the indicators of "Filter" are better than those for the "Raw" method.

From Table 3 and Table 4, we know that:

(1) Compared to the *matrix decomposition* model DisMult, the proposed GAATs model uses the neural network to mine

TABLE 3. Experimental results on the MN18RR test set. Hit@N values in percentage. The best score is in bold and second-best score is underlined.

Datasets	WN18RR									
Metric	Raw					Filter				
	MR	MRR	Hit@N			MR	MRR	Hit@N		
			@1	@3	@10			@1	@3	@10
DisMult[19]	7020	0.414	41.2	48	49.5	7000	0.419	42.3	49.6	51.2
ConvE[15]	4464	**0.446**	42.3	47.2	54	4412	0.455	42.2	48.2	54.3
CapsE[18]	**1227**	0.415	33.7	46.2	56	**1200**	0.428	6.02	46.8	57
TransE[5]	2300	0.239	4.30	43.8	52.9	2289	0.247	4.50	44.2	53.1
TransH[7]	2179	0.256	4.90	45.2	53.7	2126	0.279	5.31	46.3	54
ConvKB[16]	1322	0.270	5.82	44.5	56.0	1276	0.292	6.21	45.7	57.3
R-GCN[21]	6700	0.135	20.7	13.9	9	6254	0.148	23.4	15.9	9.9
n-hop-GATs[14]	1940	0.440	36.1	48.3	58.1	1900	**0.475**	37.6	48.9	59.0
Our work	1315	0.440	**42.5**	**50.3**	**59.6**	1270	0.467	**42.4**	**52.5**	**60.4**

TABLE 4. Experimental results on the FB15k-237 test set. Hit@N values in percentage. The best score is in bold and second-best score is underlined.

Datasets	FB15k-237									
Metric	Raw					Filter				
	MR	MRR	Hit@N			MR	MRR	Hit@N		
			@1	@3	@10			@1	@3	@10
DisMult[19]	521	0.295	21.0	31.9	46.1	500	0.310	22.1	33.8	49.2
ConvE[15]	252	0.335	23.2	35.2	50.3	241	0.348	23.8	36.9	52.3
CapsE[18]	303	0.523	48.9	50.6	59.3	282	0.541	50.0	53.0	60.9
TransE[5]	347	0.294	20.4	38.1	46.5	317	0.307	22.7	40.7	47.6
TransH[7]	234	0.314	28.4	42.6	58.5	219	0.320	30.6	45.0	61.3
ConvKB[16]	239	0.289	20.0	32.5	47.3	227	0.304	23.0	34.7	48.7
R-GCN[21]	600	0.164	11	18.0	29.4	540	0.216	13.4	13.5	30.9
n-hop-GATs[14]	219	0.508	45.9	54.1	60.4	199	0.533	47.9	54.0	61.9
Our work	**203**	**0.525**	**49.4**	**55.0**	**63.7**	**187**	**0.547**	**51.2**	**57.2**	**65.0**

the deep features of the triples to extract more feature information in the link prediction task. Therefore, all indicators for the proposed method are higher than the matrix decomposition method;

(2) Compared to the *translational model* (TransE, TransH), since the core of the translational model is $h + r \approx t$ for golden triples, which ignores the features of the triple itself, the proposed GAATs model emphasizes $h + r \approx t$ with adding attention mechanisms. The graph model has been introduced under the premise of the neural network, and the n-hop neighbors of the triples have been found, which enriches the meaning of the triple. Consequently, the experimental results for the proposed model are better than the *translation models*.

(3) Compared to the *neural network models* (ConvE, ConvKB, R-GCN, CapsE), the proposed GAATs model applies the encoder-decoder model, which uses two different neural network structures to describe the features of the triples. In particular, CapsE acts as a decoder to discover the hidden features of the triples so that the proposed model performs better than the *neural network models*.

(4) Finally, compared to the n-hop-GATs, the proposed GAATs model introduces an attenuated attention mechanism, which emphasizes the closer the relation to a given entity is, the higher the attention weight obtains. In addition, we replace the ConvKB with CapsE, so that the proposed model performance is also better.

The results also show that the performance on the WN18RR dataset is lower than on the FB15k-237 dataset. The main reason is that there are fewer types of relations on WN18RR and there are fewer intermediate paths in the relation, which can not take advantage of our model.

For each relation r on FB15k-237, we calculate the average number n_h of head entities per tail entity and the average number n_t of tail entities per head entity [24]. If $n_h < 1.5\&\&n_t < 1.5$, r is classified as one-to-one (1-1). If $n_h < 1.5\&\&n_t \geq 1.5$, r is classified as one-to-many (1-M). If $n_h \geq 1.5\&\&n_t < 1.5$, r is classified as many-to-one (M-1). If $n_h \geq 1.5\&\&n_t \geq 1.5$, r is classified as many-to-many (M-M).

Figure 7 shows Hits@10 and MRR results for predicting the head and tail entities of each relation category on FB15k-237. The experimental results show that our model in "side1" (i.e., predicting head entities in 1-1 and 1-M; predicting tail entities in 1-1 and M-1) and "sideM" (i.e., predicting head entities in M-1 and M-M; predicted tail entities in 1-M and M-M) have good performance. The main reasons are: (1) We provide an embedding representation for each triple, making the representation of the triple more precise, and the discrimination between the triples is higher. so compared to other models, our model has better performance; (2) The proposed attenuated attention mechanism makes the relation learned by each triple differently, so when making predictions, whether (r_k, e_j) or (e_i, r_k), they can still learn different representations,

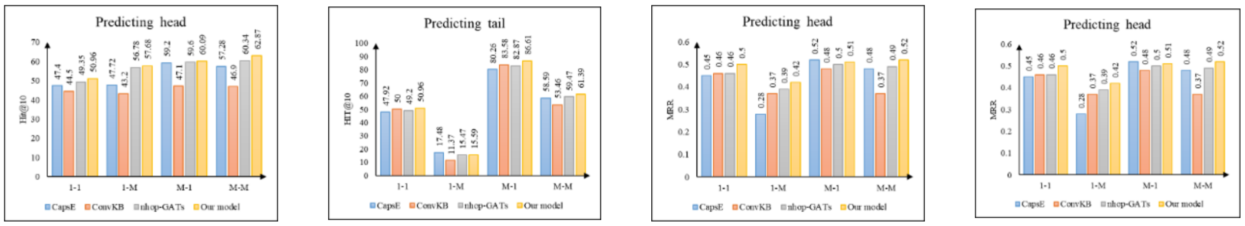


FIGURE 7. Hit@10 (in percentage) and MRR on the FB15k-237 test set w.r.t each relation category.

TABLE 5. Hit@10 on the WN18RR validation set with different *n* and iteration values. The best score is in bold.

n	100	200	300	400	500	1000	1200
1	15.9	20.7	23.1	29.4	35.7	42.9	39.8
2	24.9	32.5	36.7	39.7	45.6	50.8	48.1
3	32.1	**35.8**	44.1	**47.9**	**50.7**	**59.6**	**56.7**
4	**33.2**	35.4	**45.1**	46.7	50.4	57.6	54.3

so the results are more accurate when ranking. (3) The n-hop neighbors of a given triple in the encoder of our model replaces the embedding representation of the triple itself so that the triple carries more features and the decoder applies the CapsE instead of max-pooling ConvKB to ensures that the triple features are not lost. Based on this, the generalization ability of the model is further improved.

Figure 8 shows the results for Hits@10 and MRR on WN18RR. *also_see*, *similar_to*, *derivationally_related_form*, *verb_group* can all be regarded as M-M relations, and our model performs better than others.

Meanwhile, we study the value of n-hop neighbors and the effect of the number of iterations on the final result. Table 5 shows the value of the validation set Hit@10 of WN18RR when the number of iterations and the value of n change. When $n = 3$ and epochs $= 1000$, the model works best. We can see that as the training progresses, our model collects more information from the neighbors, which focuses more on the direct neighbor and gets secondary information from the far neighbor. Once the model converges, it learns to collect multi-hop and cluster relation information from the node n-hop neighbors. However, when $n > 3$, the effect of the model will decrease. This also implies that the farther away from the relation, the lower the attention value obtained. When $n = 4$ or more, the neighbor nodes are no longer important.

E. RELATION PREDICTION

Relation prediction requires two steps: the discovery of a pair of entities with potential relation, and the reasoning of potential relations.Our experiment assumes that an entity pair with a potential relation has been recognized. i.e., given (e_i, e_j) to predict r_k.

We drop the relations of the triples in the test data, reasoning the relations based on the embeddings of the trained entities and relations, and comparing it with the standard answer to calculate the accuracy. Besides, we apply Precision, Recall, and F-Measure in machine learning algorithms

For each pair of entities, we traverse all the relations into a triple. For each triple, we calculate the distance d betweent the embedding vector from the head entity and the relation to the tail entity. The smaller the distance $d(h + r,t)$, the greater the possibility that the triple is established.We record the relations of top 10 triples for each pair of entities in ascending order of the distance $d(h + r,t)$. We record the Hit@10 and Hit@1 of the correct relations as the accuracy rate, and record the recall rate and calculate the F-Measure. The experimental results are shown in Table 6 and Table 7.

From Table 6 and Table 7, we know that:

(1) On the WN18RR and FB15k-237 datasets, our model has achieved good result in relation prediction. Because there are only 11 kinds relations in WN18RR, and FB15k-237 contains 237 kinds of relations, the model has better effect on simple dataset WN18RR than the complex one FB15k-237.

(2) Since our model has independently embedded of relations and incorporates the distance information of the relations in the entity embedding, our model is better than the state-of-the-art models in most cases. Among them, the translational models(TransE/TransH) perform the worst. Because they only models the relations by optimizing $h + r \approx t$, and obviously ignores the semantic information of the relations. Our model adds an attenuated attention mechanism based on n-hop-GATs, so that each relation can obtain different weights according to the distance of the path, and solve the probem that n-hop-GATs does not separately embed the relations. Our model uses CapsE as a decoder and uses GAATs as an encoder to further mine the semantic information of the relations to prove the performance of relation prediction.

(3) The experimental results also show the importance of separate embedding of relations in relation prediction. In order to ensure the accuracy of the relation prediction, the independent effect of the relationship embedding is far better than the auxiliary information embedded only as the entity.

F. TRIPLE CLASSIFICATION

The task of triple classification is to determine whether a given triple (e_i, r_k, e_j) is a golden triple or not, which is a binary classification task. We use the WN18RR and FB15k-237 datasets for these experiments as well. We create two

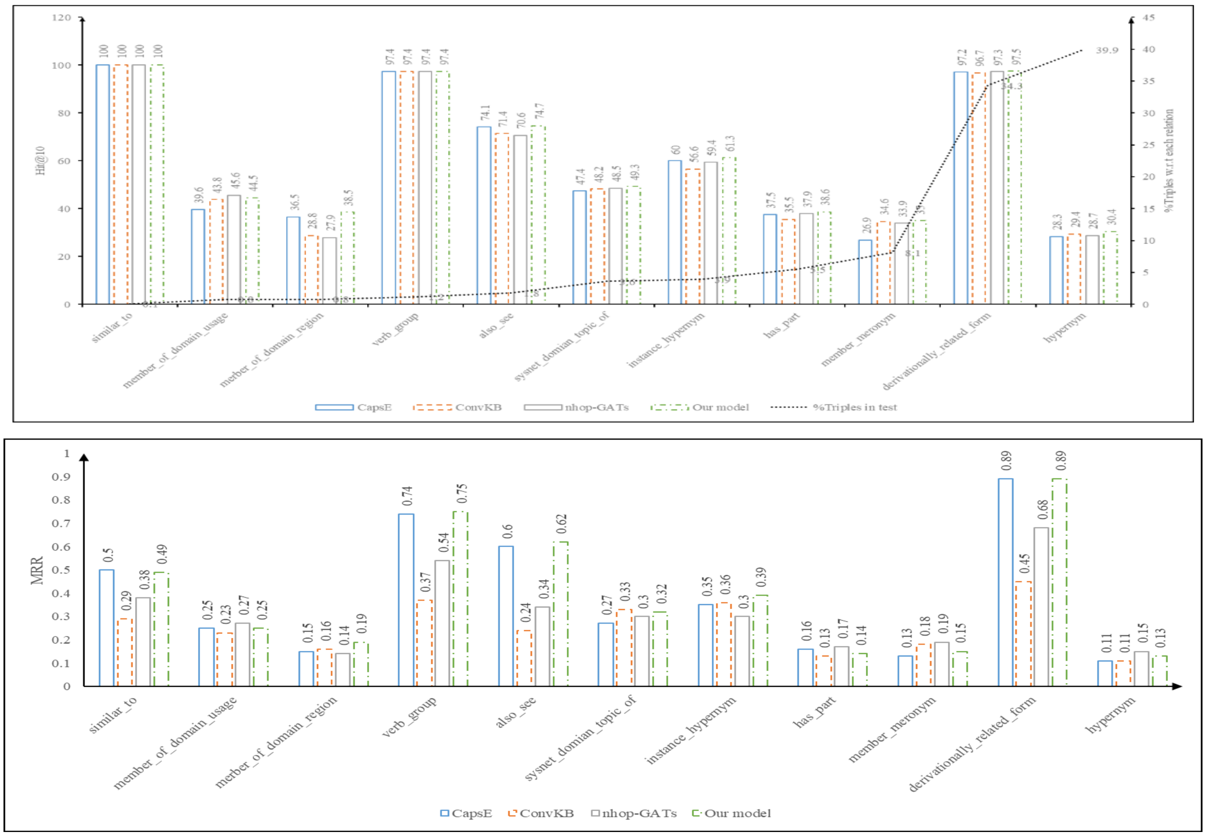


FIGURE 8. Hit@10 (in percentage) and MRR on the WN18RR test set w.r.t each relation category. The right y-axis is the percentage of triples corresponding relations.

TABLE 6. Experimental results on the MN18RR test set. The best score is in bold and second-best score is underlined.

Dataset	WN18RR					
Models	Hit@10_precision(%)	Hit@10_recall(%)	Hit@10_F(%)	Hit@1_precision(%)	Hit@1_recall(%)	Hit@1_F(%)
DisMult[19]	83.71	60.49	70.23	68.74	55.09	61.16
TransE[5]	79.82	50.27	61.69	73.40	40.09	51.86
TransH[7]	82.30	51.09	63.04	75.41	42.36	52.25
ConvKB[16]	85.50	68.31	75.94	80.24	61.90	69.89
R-GCN[21]	83.20	61.29	70.58	79.76	63.00	70.40
n-hop-GATs[14]	89.09	72.81	80.13	79.86	63.52	70.76
CapsE[18]	**89.97**	<u>73.25</u>	<u>80.75</u>	<u>83.45</u>	<u>63.26</u>	<u>71.97</u>
Our work	<u>89.86</u>	**78.56**	**83.83**	**84.72**	**69.13**	**76.14**

TABLE 7. Experimental results on the FB15k-237 test set. The best score is in bold and second-best score is underlined.

Dataset	FB15k-237					
Models	Hit@10_precision(%)	Hit@10_recall(%)	Hit@10_F(%)	Hit@1_precision(%)	Hit@1_recall(%)	Hit@1_F(%)
DisMult[19]	62.82	46.98	54.25	53.34	40.65	46.14
CapsE[18]	**68.03**	<u>56.96</u>	<u>62.00</u>	**59.70**	44.05	50.69
TransE[5]	53.54	39.58	45.44	46.50	37.26	41.37
TransH[7]	57.69	42.08	48.66	48.21	39.81	43.61
ConvKB[16]	67.05	51.24	58.09	56.91	44.85	50.17
R-GCN[21]	65.69	49.58	56.51	54.31	42.90	47.94
n-hop-GATs[14]	<u>67.21</u>	53.90	59.82	58.36	<u>47.00</u>	<u>52.07</u>
Our work	66.97	**60.21**	**63.90**	<u>59.09</u>	**52.48**	**55.56**

corrupted triple sets as presented in Section IV.B for the triple classification experiment.

For the triple classification task, we set a threshold δ_r for each relation, which is obtained by maximizing the classification accuracy of the validation set. Given a triple (e_i, r_k, e_j), if its score is greater than δ_r, it is classified as a golden triple, otherwise it is classified as corrupted [25].

We choose TransE, TransH, TransR, TransD, ConvKB, and n-hop-GATs as our baseline methods. In this experiment, we select ADADELTA SGD as our optimization goal. We

TABLE 8. Experimental results of Triple Classification (in percentage).

Datasets	WN18RR	FB15k-237
TransE[5]	75.9	79.8
TransH[7]	79.8	83.3
TransR[8]	85.9	82.1
TransD[9]	86.4	87.1
ConvKB[16]	87.1	88.4
nhop-GATs[14]	86.5	87.9
Our proposed model GAATs	**89.6**	**89.9**

choose the margin $\gamma = \{1, 2, 5, 10\}$, entity and relation embedding dimensions $k = \{50, 100, 150, 200\}$, and mini-batch size $B = \{100, 200, 500, 1000\}$. After our experiments, the optimal parameters of the validation set are $k = 100$, $\gamma = 2$, $B = 1000$ on WN18RR; $k = 100$, $\gamma = 1$, $B = 100$ on FB15k-237.

For the two datasets, we testify the task by using 1000 iterations. The experimental results are presented in Table 8. Table 8 shows that in WN18RR and FB15k-237, the accuracy of the triple classification of the proposed GAATs model is higher than the other models. The main reasons are: (1) The translational models (TransE, TransH, TransR, TransD) first randomly initialize the representation of entities and relations, and then represent the embedding of entities and relations based on $h + r \approx t$. The initial embedding is further optimized using a neural network, so our model representation is better than the translational models; (2) ConvKB method uses a CNNs to extract the features of the triple, as a decoder in the n-hop-GATs model. Although the hidden features of the triples can be found, it is easy to lose features using the pooling method. (3) Compared with the n-hop-GATs model, the proposed GAATs model first introduces the attenuated attention mechanism, and then the decoder adopts the more advanced CapsE model, so the accuracy is further improved.

V. CONCLUSION

We propose the GAATs model, which applies an attenuated attention mechanism into the representation of relations and entities to obtain new embeddings on both entities and relations. In addition, we use the CapsE as our decoder, avoiding the loss of features exhibited by the ConvKB. Therefore, our model takes the diversity of triples into account. An extensive experiment shows that our method is more effective than the state-of-the-art methods in link prediction, triple classification and relation prediction.

However, our model has a high computational complexity due to the graph feature extraction, and time cost arising from representing each triple separately. How to improve the efficiency of the graph feature extraction is an urgent problem to be solved. In addition, the temporal and spatial attributes for dynamic KGs also need to be studied as the next research goal.

REFERENCES

[1] X. Fu, X. Ren, O. J. Mengshoel, and X. Wu, "Stochastic optimization for market return prediction using financial knowledge graph," in *Proc. IEEE Int. Conf. Big Knowl. (ICBK)*, Nov. 2018, pp. 25–32.

[2] M. Rotmensch, Y. Halpern, and A. Tlimat, "Learning a health knowledge graph from electronic medical records," *Sci. Rep.*, vol. 7, no. 1, p. 5994, 2017.

[3] C. Xiong, R. Power, and J. Callan, "Explicit semantic ranking for academic search via knowledge graph embedding," in *Proc. 26th Int. Conf. World Wide Web (WWW)*, 2017, pp. 1271–1279.

[4] R. Socher, D. Chen, and C. D. Manning, "Reasoning with neural tensor networks for knowledge base completion," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 926–934.

[5] A. Bordes, N. Usunier, and A. Garcia-Duran, "Translating embeddings for modeling multi-relational data," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2013, pp. 2787–2795.

[6] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proc. ICML*, vol. 11, 2011, pp. 809–816.

[7] Z. Wang, J. Zhang, and J. Feng, "Knowledge graph embedding by translating on hyperplanes," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 1112–1119.

[8] Y. Lin, Z. Liu, and M. Sun, "Learning entity and relation embeddings for knowledge graph completion," in *Proc. 39th AAAI Conf. Artif. Intell.*, 2015, pp. 2181–2187.

[9] G. Ji, S. He, and L. Xu, "Knowledge graph embedding via dynamic mapping matrix," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process.*, vol. 1, 2015, pp. 687–696.

[10] H. Xiao, M. Huang, Y. Hao, and X. Zhu, "TransA: An adaptive approach for knowledge graph embedding," Sep. 2015, *arXiv:1509.05490*. [Online]. Available: https://arxiv.org/abs/1509.05490

[11] T. N. Kipf and M. Welling, "Semisupervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017.

[12] M. Fan, Q. Zhou, and E. Chang, "Transition-based knowledge graph embedding with relational mapping properties," in *Proc. 28th Pacific Asia Conf. Lang., Inf. Comput.*, 2014, pp. 328–337.

[13] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[14] D. Nathani, J. Chauhan, C. Sharma, and M. Kaul, "Learning attention-based embeddings for relation prediction in knowledge graphs," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 4710–4723.

[15] T. Dettmers, P. Minervini, and P. Stenetorp, "Convolutional 2D knowledge graph embeddings," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018.

[16] D. Q. Nguyen, T. D. Nguyen, and D. Q. Nguyen, "A novel embedding model for knowledge base completion based on convolutional neural network," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2017, pp. 327–333.

[17] M. Nickel, L. Rosasco, and T. Poggio, "Holographic embeddings of knowledge graphs," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016.

[18] D. Q. Nguyen, T. Vu, T. D. Nguyen, D. Q. Nguyen, and D. Phung, "A capsule network-based embedding model for knowledge graph completion and search personalization," in *Proc. Conf. North*, 2019.

[19] B. Yang, W. Yih, and X. He, "Embedding entities and relations for learning and inference in knowledge bases," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014.

[20] Y. Lin, Z. Liu, H. Luan, M. Sun, S. Rao, and S. Liu, "Modeling relation paths for representation learning of knowledge bases," in *Proc. EMNLP*, 2015.

[21] M. Schlichtkrull, T. N. Kipf, and P. Bloem, "Modeling relational data with graph convolutional networks," in *Proc. Eur. Semantic Web Conf.* Cham, Switzerland: Springer, 2018, pp. 593–607.

[22] A. Vaswani, N. Shazeer, and N. Parmar, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

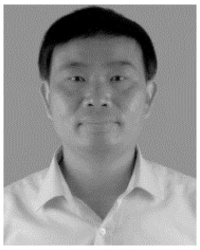
[23] D. Q. Nguyen, "An overview of embedding models of entities and relationships for knowledge base completion," Mar. 2017, *arXiv:1703.08098*. [Online]. Available: https://arxiv.org/abs/1703.08098

[24] D. Q. Nguyen, K. Sirts, L. Qu, and M. Johnson, "STransE: A novel embedding model of entities and relationships in knowledge bases," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2016, pp. 460–466.

[25] C. Xu and R. Li, "Relation embedding with dihedral group in knowledge graph," Jun. 2019, *arXiv:1906.00687*. [Online]. Available: https://arxiv.org/abs/1906.00687
[26] S. Keizer, M. Guhe, H. Cuayáhuitl, I. Efstathiou, K.-P. Engelbrecht, M. Dobre, A. Lascarides, and O. Lemon, "Evaluating persuasion strategies and deep reinforcement learning methods for negotiation dialogue agents," in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics*, vol. 2, 2017, pp. 480–484.
[27] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon, "Representing text for joint embedding of text and knowledge bases," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 1499–1509.
[28] X. V. Lin, R. Socher, and C. Xiong, "Multi–hop knowledge graph reasoning with reward shaping," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018.
[29] M. Sarwar and M. Akram, "M-polar fuzzy concept lattices with applications," *New Math. Natural Comput.*, vol. 13, no. 3, pp. 261–287,2017.
[30] M. Sarwar and M. A. Bipolar, "Fuzzy circuits with applications," *J. Intell. Fuzzy Syst.*, vol. 34, no. 1, pp. 547–548, 2018.
[31] X. Lin, Y. Liang, and F. Giunchiglia, "Relation path embedding in knowledge graphs," *Neural Comput. Appl.*, vol. 31, no. 9, pp. 5629–5639, 2019.
[32] Y. Shen, D. Wen, and Y. Li, "Path-based attribute-aware representation learning for relation prediction," in *Proc. SIAM Int. Conf. Data Mining. Soc. Ind. Appl. Math.*, 2019, pp. 639–647.
[33] M. Zhang, Q. Wang, and W. Xu, "Discriminative path-based knowledge graph embedding for precise link prediction," in *Proc. Eur. Conf. Inf. Retr.* Cham, Switzerland: Springer, 2018, pp. 276–288.
[34] X. Lv, Y. Gu, X. Han, L. Hou, J. Li, and Z. Liu, "Adapting meta knowledge graph information for multi–hop reasoning over few–shot relations," in *Proc. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 3367–3372.
[35] F. Ma, F. Gao, J. Sun, H. Zhou, and A. Hussain, "Attention graph convolution network for image segmentation in big SAR imagery data," *Remote Sens.*, vol. 11, no. 21, p. 2586, Nov. 2019.
[36] J. B. Lee, R. Rossi, and X. Kong, "Graph classification using structural attention," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 1666–1674.
[37] H. Zhou, T. Young, M. Huang, H. Zhao, J. Xu, and X. Zhu, "Commonsense Knowledge Aware Conversation Generation with Graph Attention," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 4623–4629.
[38] X. Wang, D. Wang, and C. Xu, "Explainable reasoning over knowledge graphs for recommendation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Aug. 2019, pp. 5329–5336.
[39] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "KGAT: Knowledge graph attention network for recommendation," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2019.
[40] K. Guo, D. Wang, J. Huang, Y. Chen, Z. Zhu, and J. Zheng, "A graph representation learning algorithm based on attention mechanism and node similarity," in *Proc. CCF Conf. Comput. Supported Cooperat. Work Social Comput.*, 2019, pp. 591–604.
[41] S. Abu-El-Haija, B. Perozzi, and R. Al-Rfou, "Watch your step: Learning node embeddings via graph attention," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 9180–9190.

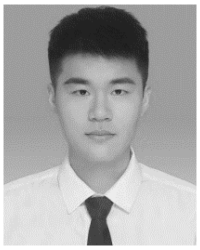
RUI WANG received the B.E. degree from the Department of Computer Science and Engineering, Hunan University of Science and Technology, Hunan, China, in 2017. He is currently pursuing the master's degree with the College of Computer Science and Technology, Huaqiao University.

His main research areas are the construction of knowledge graphs, including entity disambiguation and knowledge reasoning, and so on.



BICHENG LI received Ph.D. degree from the National University of Defense Technology, in June 1998.

From September 2004 to September 2005, he was a Visiting Scholar with the University of Manchester, U.K. From June 1998 to April 2016, he was a Lecturer with the Information Engineering University. In May 2016, he was introduced as a Leader of the discipline, a Distinguished Professor, and the Master of Huaqiao University. His main research directions include: intelligent information processing, network ideology security, network public opinion monitoring and guidance, and big data analysis and mining.



SHENGWEI HU received the B.E. degree from the School of Software, Jiangxi Normal University of Science and Technology, Jiangxi, in 2017. He is currently pursuing master's degree with the College of Computer Science and Technology, Huaqiao University. His research areas are natural language processing and computational linguistics.

WENQIAN DU received the B.E. degree in computer science and technology from Fuyang Normal University, Fuyang, China, in 2018. She is currently pursuing the master's degree in computer technology with Huaqiao University. Her research interests include knowledge graph inference and knowledge embedding.

MIN ZHANG received the B.E. degree from the Department of Information Management and Information System, Shandong University of Technology, Shandong, China, in 2019. He is currently pursuing the master's degree with the College of Computer Science and Technology, Huaqiao University.

His main research area is the construction of knowledge graphs, including event extraction and knowledge reasoning.