IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 27, 2020, accepted December 7, 2020, date of publication December 14, 2020,
date of current version December 29, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3044343

Graph Attention Networks With Local Structure Awareness for Knowledge Graph Completion

KEXI JI[1], BEI HUI[1,2], AND GUANGCHUN LUO[1,2], (Member, IEEE)
[1]School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China
[2]Trusted Cloud Computing and Big Data Key Laboratory of Sichuan Province, Chengdu 610054, China

Corresponding author: Bei Hui (bhui@uestc.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFC0807500, in part by the National Natural Science Foundation of China under Grant U19A2059, and in part by the Ministry of Science and Technology of Sichuan Province Program under Grant 2018GZDZX0048 and Grant 20ZDYF0343.

ABSTRACT Graph neural networks have been proven to be very effective for representation learning of knowledge graphs. Recent methods such as SACN and CompGCN, have achieved the most advanced results in knowledge graph completion. However, previous efforts mostly rely on localized first-order approximations of spectral graph convolutions or first-order neighborhoods, ignoring the abundant local structures like cycles and stars. Therefore, the diverse semantic information beneath these structures is not well-captured, leaving opportunities for better knowledge representation which will finally help KGC. In this work, we propose LSA-GAT, a graph attention network with a novel neighborhood aggregation strategy for knowledge graph completion. The model can take special local structures into account, and derive a sophisticated representation covering both the semantic and structural information. Moreover, the LSA-GAT model is combined with a CNN-based decoder to form an encoder-decoder framework with a carefully designed training process. The experimental results show significant improvement of the proposed LSA-GAT compared to current state-of-the-art methods on FB15k-237 and WN18RR datasets.

INDEX TERMS Knowledge graph embedding, knowledge graph completion, graph attention networks.

I. INTRODUCTION

Knowledge Graphs (KGs), such as Freebase [1], YAGO [2], DBpedia [3] and NELL [4] are fashionable carriers for various common-sense knowledge. They act as the core of many state-of-the-art natural language processing solutions to many practical applications, including question answering, reading comprehension, *etc*. However, the fact, where knowledge graph is usually incomplete and lacks facts, entities, or relations, may significantly hinder the performance [5]. Therefore, the knowledge graph completion is necessary and vital. Figure 1 shows a subgraph of a knowledge graph with real facts composed of actual relations between two entities (solid lines). The relationships are organized in the forms of (h, r, t) triples (e.g. h = *"Tom Hardy"*, r = *"acted_in"*, t = *"Revenant"*). The task of knowledge graph completion is aimed at reasoning over existing triples to find the missing links (e.g. red dotted line in Figure 1).

Current knowledge graph completion methods are broadly divided into four categories: Translation-based models, Tensor Factorization models, CNN-based models, and Graph Neural Network models. The translation-based models

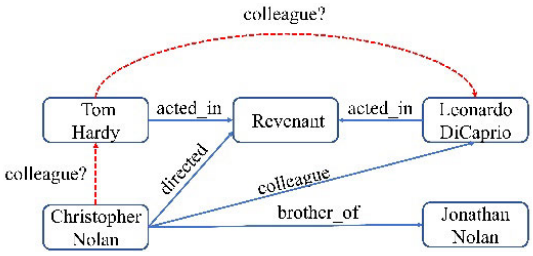


FIGURE 1. A subgraph in a Knowledge Graph in representation of triples which is composed of actual relations (solid lines) between entities and inferred relations (red dotted line).

[6]–[8], tensor factorization models [9]–[12], and CNN-based models [13]–[15] treat each triple independently, and try to model different relations to capture the semantic information of the knowledge graph. However, they fail to capture the rich sematic presented near a given entity in a KG. Therefore, models based on graph neural networks (GNNs) become fashionable recently. These GNN models explicitly use neighboring information of an entity.

However, current methods based on GNNs [23] cannot make good use of the rich local structural information in the knowledge graph. For example, in Fig.2, there are similar contexts around entity *Hermione Granger* and *Draco*

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

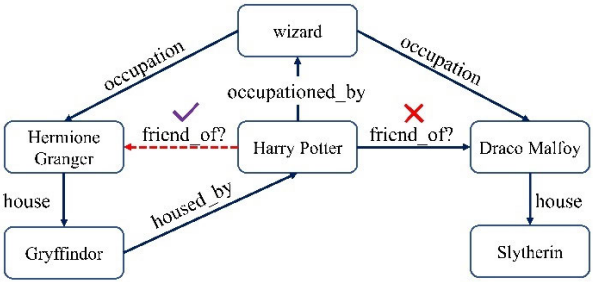


FIGURE 2. An example of subgraph. There are similar contexts around entity Hermione Granger and Draco Malfoy which have different local structure, (i.e. relation paths (house, Houseed_by), (occupation, occupationed_by) for Hermione Granger, and only (occupation, occupationed_by) for Draco Malfoy.

Malfoy for them having the same neighbor *wizard* and *Harry Potter*. Current GNN models are likely to predict that *(Harry Potter, friend_of, Hermione Granger)* and *(Harry Potter, friend_of, Draco Malfoy)*. However, two head entities *Hermione Granger* and *Draco Malfoy* actually have different local structures, i.e., relation paths *(house, housed_by)*, *(occupation, occupationed_by)* for Hermione Granger, and only *(occupation, occupationed_by)* for Draco Malfoy. Applying such local structures may benefits the completion of knowledge graphs, which are not properly handled by current methods like basic GAT models.

Therefore, for the comprehensive usage of such structural information in knowledge graph completion, we propose local structure-aware graph attention networks (LSA-GAT). The model applies the local structure around an entity and aggregate the local structural information to update entities and relations' embeddings. Our model first observes multiple typical structures near the given entity like stars and triangles, then aggregates different structure information to learn different embeddings for the entity, and finally combines these different embedding representations through the attention coefficient to obtain the updated representation for the entity.

Through this design, our proposed model can capture the potential semantic logics between entities and relations within the representation of entities and relations. We further integrate the model with a CNN-based module to compose an encoder-decoder framework. Through experiments, we have proved our proposed model is superior to the existing GNN-based model in learning graph structure information, and the experimental effect shows strong competitiveness. Our contributions are as follows:

- As far as we know, this is the first study to propose a graph attention method that explicitly uses different graph local structures for knowledge graph completion.
- We proposed an end-to-end network LSA-GAT taking benefits of both GAT and CNN entities and relations, and the decoder applies 3×3 convolution filters and 1×3 filters to capture the interactive and translational characteristic between entities and relations.
- We demonstrate the effectiveness of our proposed LSA-GAT on the standard FB15K-237 and WN18RR datasets, and show about a mean 10% relative improvement over the state-of-the-art SACN in terms of Hits@1, Hits@3 and Hits@10.
- Extensive experimental results reveal that different local structures contribute diversely to link prediction performance, and the overall performance is proportional to the number of entity local structures applied, which validate that local structure information is meaningful for knowledge graph completion.

II. RELATED WORK

Recently, Knowledge Graph Completion (KGC) or Link Prediction (LP), aimed at inferring missing head entities, tail entities or relations in triples, has become an active research field. The earliest model is based on the embedding method, and its main idea is to model relation with different properties such as symmetric and asymmetric relations in the graph. TransE [6] models the triples as the translation of the head entity to the tail entity in the semantic space, but it can only handle 1-1 relations. To solve this question, TransH [7] is proposed to deal with 1-N relations by projecting head and tail entities into a relation-specific hyperplane and then computed the triple score using a translation operation on the hyperplane. TransR [8] directly build separate relations and entities space, projecting entities from entity space to relation-specific space to calculate the distance between entities.

However, researchers found that when the scale of the knowledge graph continues to grow, the above translation-based model cannot achieve a balance between the expansion ability and expression power. Tensor Factorization models are proposed to compress the model. RESCAL [9] applies a three-way rank-r factorization over each relational slice of knowledge graph tensor. DistMult [10] learns embeddings from a bilinear objective and captures relational semantics using weighted elementwise dot products to model entity relations. ComplEX [11] generalizes DistMult [10] by using complex embeddings and Hermitian dot products instead. TuckER [12] learns embedding by outputting a core tensor and embedding vectors of entities and relations.

Recently, CNN-based models have been proven to improve expression ability by capturing complex interactions between entities and relations. At the same time, the parameter efficiency of CNN can prevent the model from becoming difficult to run as the scale of the knowledge graph expands. ConvE [13] reshapes the head entity and relation embeddings into a two-dimensional matrix and then applies a 2D convolution and a fully connected layer on top of it to obtain a feature vector. This feature vector and the tail entity embedding vector are thrown to an inner product layer for the final prediction. Just like the application of the capsule network in the image processing field, CapsE [14] captures the complex high-level features in the triples by applying a capsule network after the convolutional layer. InteractE [15] increases the interaction between relation and entity embeddings through three key ideas—feature permutation, a novel feature reshaping, and circular convolution. And its

experiments show increasing the number of such interactions is beneficial to link prediction performance. Although CNN-based methods are effective, such methods always treat each triplet independently and cannot capture the connections between the triples. Moreover, methods above all only treat each triple independently, ignoring the rich contextual information around the triple. To handle this, GNN-based models are proposed. GNNs are introduced for learning connectivity structure under an encoder-decoder framework. R-GCN [16] proposes relation-specific transformation to model the directed nature of knowledge graphs. SACN [17] proposed weighted GCN, defining the strength of two adjacent nodes with the same relation type, to capture the structural information in knowledge graphs by utilizing node attributes, and relation types. CompGCN [18] introduces a novel Graph Convolutional framework which jointly embeds both nodes and relations in a relational graph.

Existing methods mostly treat triples independently or aggregate neighbors without take some special local structures into account. However, sometimes these neglected local structures play an important role [24]. ADSF [25] uses random walks to find subgraphs around nodes, and learns node representations by fusing the information of these subgraphs instead of separate neighbor node information to help complete the classification of nodes on citation networks and social networks. In comparison, our proposed model LSA-GAT aggregates neighbors with local structure awareness.

III. METHOD

This section first introduces the basic definitions and notations of KGC, and preliminaries on the pure graph attention strategy for neighbor node aggregation. Then the motivation and overview of our model are given. The third part provides the intuition, the general structure, and the detailed component design of the proposed end-to-end model LSA-GAT.

A. PROBLEM FORMULATION AND NOTATION

Because of the nature of incompleteness of knowledge graphs, KGC is developed to add new triples to a knowledge graph. Here gives a task-oriented definition.

Definition 1 (Knowledge Graph Completion): Given an incomplete knowledge graph $G = (E, R, F)$, in which E is the set of entities, R is the set of existed factual triples in this knowledge graph. Knowledge graph completion (KGC) aims to find missing facts by reasoning over existed triples. Figure 1 shows an example of knowledge graph completion, based on existed triples, we can infer missing links. For instance, we can infer *(Tom Hardy, colleague, Leonardo DiCaprio)* (red dotted line) by reasoning over existed triples (solid lines).

1) PRELIMINARY (GRAPH ATTENTION NETWORKS FOR AGGREAGATING NEIGHBORS)

GAT assighn different weights for different neighbor around an entity and recursively learns the representation for entities and relations. Formally, at each layer l of GAT, entity i integrates neighboring entities' features to obtain a new representation via:

$$h^{(l+1)} = \sigma(\sum_{j \in N_i} \alpha_{ij} h_j^l w^l) \tag{1}$$

where h^l is the representation of entity i at l-th layer; N_i is the set of entity i's neighbors; α_{ij} is calculated weights via:

$$\alpha_{ij} = g(h_i^l, \sum_{j \in N_i} (r_{ij}, h_j^l)) \tag{2}$$

where r_{ij} is the representation of relation between entity i and entity j, g is some attention calculation function. Figure 3 shows the aggregation process of GAT.

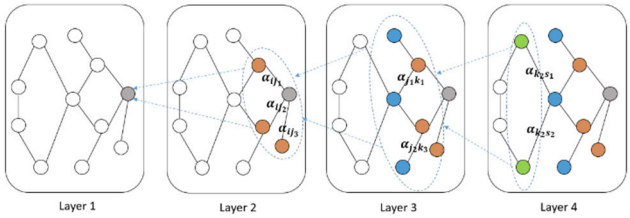


FIGURE 3. The process of GAT. In each layer, entities aggregate the information of their own first-order neighbors with GAT and update their embeddings.

We can view a GAT [27] as a message-passing algorithm. Each layer in a GAT allows the model to integrate information from a wider neighborhood. We explain this from the perspective of a target entity (in gray). In Layer 2, this entity gathered information from its one-hop neighbors (in orange). In Layer 3, orange entities integrate their own one-hop neighbors' information (in blue). In Layer 4, blue entities similarly aggregate information from their immediate neighbors (in green). However, in the entire message passing process, each triplet is only be treated as an independent individual, ignoring the association between the triplet and local structures patterns.

B. MOTIVATION AND OVERVIEW

Actually, the understanding of an entity in the knowledge graph can start from its local area. In a knowledge graph, some local structure contains rich semantic information, which can help link prediction. Figure 4(a) shows a subgraph of the entity "Ran" in FB15k-237 dataset, we can see that the local structure around the entity is mainly divided into three types: 1) connect to the same entity through different relations; 2) connect to an entity through a path and another relation at the same time; 3) connect to different entities through different relations.

Similarly, we can abstract the whole graph and define following local structures:

- 2-Cycle (in red): Entity v and another entity t are connected through two different relations. formally, there are two triples (v, r_1, e_1) and (v, r_2, e_1) in which only the relation is different.
- Triangle (in blue): Entity v and entity t are connected through a relation and a path respectively. Formally,

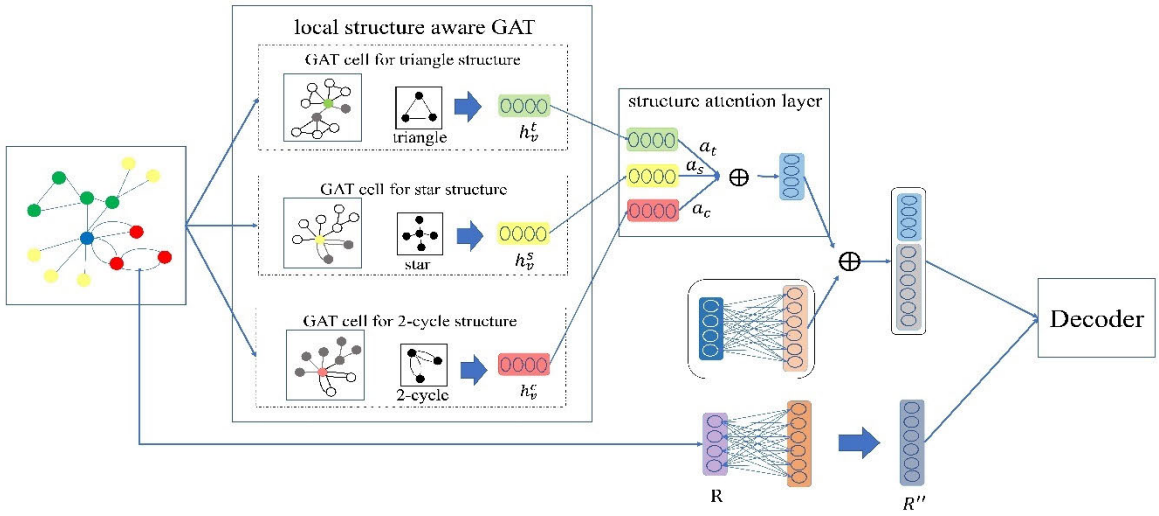


FIGURE 4. A subgraph of entity "Ran" in FB25k-237 dataset and its illustration. (a) shows a valid subgraph around entity "Ran", (b) shows its corresponding illustration under our definition.

(v, r_5, e_3) denotes the relation between entity v and entity e_3, $(v, r_3, e_2) \wedge (e_2, r_4, e_3)$ denote the path.

- Star (in green): Entity v is linked to different entities through different relations. Formally, for a given entity v, there are two triples (v, r_6, e_4), (v, r_7, e_5), (v, r_8, e_6) and (v, r_9, e_7) in which only the head entity is the same.

These three different local structures can reveal different semantic information. For example, the 2-cycle structure reveals that different relations connected to the same entity have similar semantics. To learn the structural information, we proposed a local structure-aware GAT model, as LSA-GAT. Figure 5 shows the overview of the architecture of our model.

Our proposed model mainly contains four modules: i) **Graph Attention Module**; ii) **Local Structure Representation Module**; iii) **Feature Fusion Module**; iv) **Decoder**. **Graph attention module** measures the importance of each neighbor triple around the target entity; **local structure representation module** performs different process on different local structures to abtain the structure-specific embeddings; **feature fusion module** merges the structure-specific embeddings to obtain the final embedding for entities; the module i),ii) and iii) form the **Encoder** of our model; **Decoder** module uses the learned embeddings to calculate triples' authenticity score based on an advanced CNN model.

C. ENCODER

In this section, we will give the details of the encoder in our model. The encoder mainly contains three parts: 1) graph attention mechanism; 2) representation of the local structures; 3) feature fusion and update equation.

1) GRAPH ATTENTION MECHANISM

We adopt the graph attention networks (GATs) with redesigned attention formula to recursively learn the representation for entities by aggreagating the information of neighboring triplesthe entity representation for

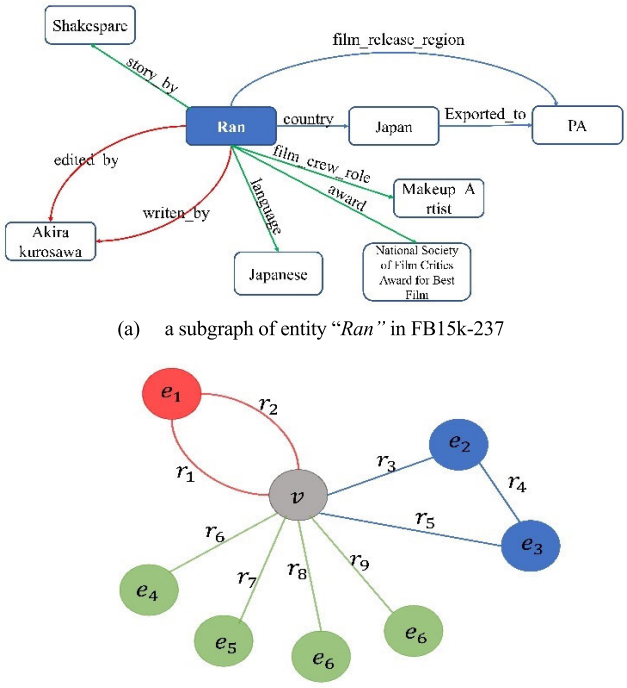


FIGURE 5. The architecture of our model. Our model contains two main parts: Encoder and Decoder. Encoder is constructed based on our proposed LSA-GAT to learn entities and relations' representations, and decoder is aimed at scoring would-be triples to infer missing facts using the learned embeddings with CNN-based method.

knowledge graphs. Figure 6 shows the design of the graph attention mechanism used in our model.

Generally, given an entity's subgraph within one hop, the entity and relations' embeddings are inputted into a graph attention cell and output the learned representations of the entity and relations. Formally, given an entity v (we call it the target entity), its' neighbors are represented by

$$N(v) = \{(v_i, r_{ij}, v_j) | r_{ij} \in R, v_j \in E, (v_i, r_{ij}, v_j) \in F$$

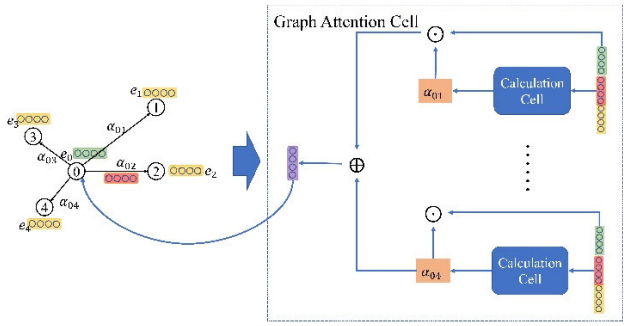


FIGURE 6. Graph attention mechanism used in our model.

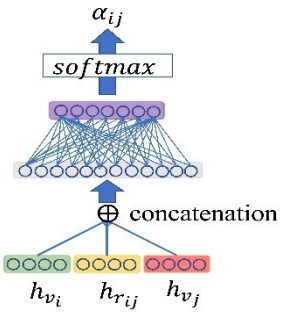


FIGURE 7. The calculation cell used in graph attention mechanism to compute weight for each neighbor.

We calculate the attention weights of a neighbor using head entity v_i and tail entity v_j as well as the relation r_{ij}. Firstly, we concatenate the entity and relation embeddings to represent the corresponding triple (v_i, r_{ij}, v_j) followed a linear transformation to learn the embedding of the triple associated with entity v_i:

$$h_{(v_i,r_{ij},v_j)} = W_1 \cdot concat(h_{v_i}, h_{r_{ij}}, h_{v_j}) \tag{3}$$

where h_{v_i}, h_{t_j}, $h_{r_{ij}}$ are the embeddings of entities v_i, v_j and relation r_{ij}, $h_{(v_i,r_{ij},t_j)}$ denotes the embedding of corresponding triple. W_1 denotes the linear transformation matrix. Then we perform LeakyRelu and Softmax to get relative attention weight for each triple, Figure 7 shows the computation of relative attention values:

$$d_{(v_i,r_{ij},v_j)} = LeakyRelu(W_2 \cdot h_{(v_i,r_{ij},v_j)}) \tag{4}$$

$$\alpha_{ij} = softmax\left(d_{(v_i,r_{ij},v_j)}\right) = \frac{\exp(d_{(v_i,r_{ij},v_j)})}{\sum_{v_t \in N(v_i)} \exp(d_{(v_i,r_{it},v_t)})} \tag{5}$$

where $N(v_i)$ denotes the neighborhood of entity v_i, r_{it} denotes the corresponding relation linking entities v_i and v_t. The transformed embedding is calculated by summing up all triples' representation weighted by their attention value, and we use multi-head attentions [20] to stabilize the process:

$$h'_{v_i} = \sigma\left(\sum_{v_j \in N(v_i)} \alpha_{ij} \cdot h_{(v_i,r_{ij},v_j)}\right) \tag{6}$$

$$\vec{h'_{v_i}} = \frac{1}{M} \sum_{K=1}^{M} \sigma\left(\sum_{v_j \in N(v_i)} \alpha_{ij}^{(k)} \cdot h_{(v_i,r_{ij},v_j)}^{(k)}\right) \tag{7}$$

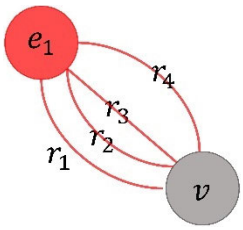


FIGURE 8. The illustration of cycle-structure.

where M is the number of attention head we used. For simplification, we denote the above attention operation as:

$$\vec{h'_{v_i}} = Att(H_{N(v)}) \tag{8}$$

where $H_{N(v)} = \{h_{(v,r_{vj},v_j)} | v_j \in N(v)\}$ is the set of embedding of neighboring triples associated with entity v.

2) REPRESENTATION OF LOCAL STRUCTURES

As mentioned in section III(B) above, to learn local structural information, we define three concerned structures, 2-cycle, triangle, and star. Then we use our proposed model LSA-GAT to aggregate the structure information to generate three different structure-specific embeddings, and finally, merge them to obtain the final embedding. For the 2-cycle structure, we form a combination relationship by combining different relationships between the same pair of entities. The combination relation is regarded as a new relation to form a 2-cycle structure triple. For the triangle structure, we perform LSTM on the path and combine it with the individual relation to form a triangle structure triple. For star-structure, we regard these triples as normal triples. Then input the reconstructed triples into different GAT cells designed in Fig.6 to learn the structure-specific embeddings for entities, and finally use simple attention to merge them up for final embeddings.

2-Cycle

For different relations connected to the same entity, we combine their relations in pairs to obtain different combination triples, which allows the model to capture the interactive characteristics between these relations.

For the example in Fig.8, there are 4 different relations between entity v and e_1. In our work, we first combine relations in pairs to get combination triples. Formally, we use combination to produce 6 combination triples $(v, r_1 + r_2, e_1)$, $(v, r_1 + r_3, e_1)$, $(v, r_1 + r_4, e_1)$, $(v, r_2 + r_3, e_1)$, $(v, r_2 + r_4, e_1)$ and $(v, r_3 + r_4, e_1)$ based on these four triples. This can capture the interactive features of different but semantic-similar relations. And then we learn the embedding of combination triples $(v, r_1 + r_2, e_1)$ (the same for another combination triples):

$$h_{(v,r_1+r_2,e_1)} = W_1 \cdot concat(h_v, h_{r_1+r_2}, h_{e_1}) \tag{9}$$

$$h_{r_1+r_2} = h_{r_1} + h_{r_2} \tag{10}$$

where h_{r_1}, h_{r_2} are the embeddings of relation r_1, r_2 respectively;h_v, h_{e_1} are the embeddings of entity v and e_1. We denote the set of combination triples as cycle-specific neighborhood $N_{(v)}^c = \{(v, r_i + r_j, e_t) | v, e_t \in E, r_i, r_j \in R_{vt}.$

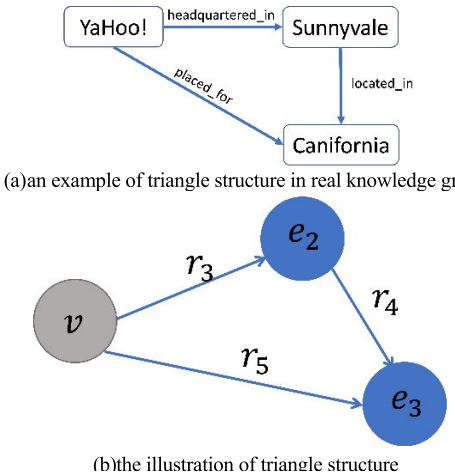


FIGURE 9. An example of triangle-structure. (a) shows an instance of triangle structure in knowledge graph, (b) shows the illustration of the triangle structure.

where R_{vt} is the set of different relation connecting entity v and e_t. And then the 2-cycle-structure-specific embedding of entities is computed by:

$$h_v^c = Att(h_{N_{(v)}^c}) \tag{11}$$

where $h_{N_{(v)}^c} = \{h_{(v,r_i+r_j,e_t)} | v, e_t \in E, r_i, r_j \in R_{vt} \mathrm{h}_{\mathrm{N}_{(\mathrm{v})}^{\mathrm{c}}} = \{\mathrm{h}_{(\mathrm{v},\mathrm{r_i}+\mathrm{r_j},\mathrm{e_t})} | v, \mathrm{e_t} \in \mathrm{E}, \mathrm{r_i}\mathrm{r_j} \in \mathrm{R_{vt}}$ is the set of embeddings for combination triples, *Att*Att is the attention mechanism introduced in (3)-(8).

Triangle:

As shown in Fig.9, (a) is an instance of triangle structure in knowledge graph, (b) shows the illustration of the triangle structure. Supposed there are a relation (v, r_5, e_3) and a 2-hop relation path $p(v, e_3) = (r_3, r_4)$, which indicates ve_2e_3, it is straightforward and reasonable for us to build path embedding via semantic composition of relation embeddings.

As illustrated in Fig.10, the path embedding p is calculated by the embedding of relation *headquatered_in*, entity *Sunnyvale* and relation *located_in*. And then we add the path embedding to the embedding of single-relation *placed_founded* to obtain a compositional relation embedding. Finally, triangle-structured triples are represented by the embeddings of head entity, compositional relation and tail entity.

Formally, consider a relation (v, r_5, e_3) and a 2-hop relation path $p(v, e_3) = (r_3, r_4)$ indicates ve_2e_3 (shown in Fig.9), we use LSTM to get the path embedding:

$$h_{lstm} = LSTM(h_{r_3}, h_{e_2}, h_{r_4}) \tag{12}$$

then the compositional relation embedding is represented as:

$$h_{cr} = h_{r_5} + h_{lstm} \tag{13}$$

the embedding of triangle is represented by following equation:

$$h_{(v,cr,t)} = W_2 \cdot concat(h_v, h_{cr}, h_{e_3}) \tag{14}$$

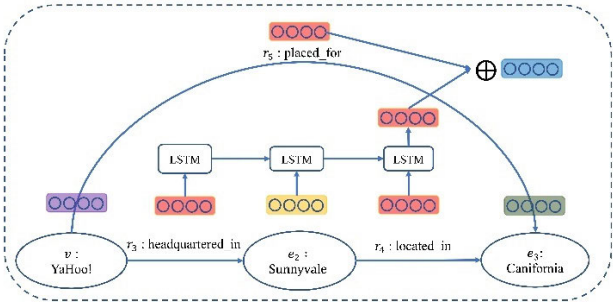


FIGURE 10. The embedding process of triangle-structured triple. We use LSTM to learn the representation of relation path, and then add the path to another single relation to get combination relation. Then the combination triple is formed of head entity, tail entity and the combination relation.

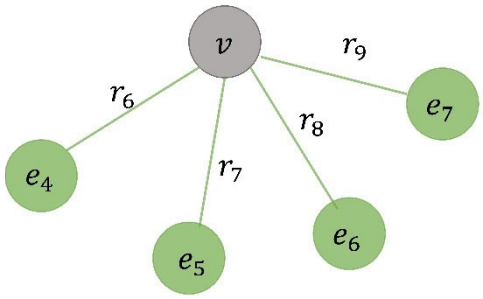


FIGURE 11. The illustration of star structure. In star structure, only the head entity stays the same.

moreover, in order to get the triangle-structured-specific embedding of the target entity, we aggregate all of surrounding triangles' information:

$$h_v^t = Att(h_{N_{(v)}^t}) \tag{15}$$

where $N_{(v)}^t$ denotes the neighboring triangles, $h_{N_{(v)}^t}$ is the embedding of these triangles, here, N denotes the number of triangles around the target entity.

Star:

The star structure reveals semantic information of entities in different aspects. That means we can regard star-structured triple as normal triple. Figure 11 shows an example of star structure.

Consider the set of star-structured triples $N_{(v)}^s = \{(v, r_j, e_t) | v, e_t \in E, r_j \in R$, the neighborhood is learned by performing $Att()$ over the embeddings of these triples:

$$h_{(v,r_j,e_j)} = W_3 \cdot concat(h_v, h_{r_j}, h_{e_j}) \tag{16}$$

$$h_v^s = Att(h_{N_{(v)}^s}) \tag{17}$$

where $h_{N_{(v)}^s} = \{h_{(v,r_j,e_j)} | v, e_t \in E, r_j \in R \mathrm{h}_{\mathrm{N}_{(\mathrm{v})}^{\mathrm{s}}} = \{\mathrm{h}_{(\mathrm{v},\mathrm{r_j},\mathrm{e_j})} | v, \mathrm{e_j} \in \mathrm{E}, \mathrm{r_j} \in R\}$.

3) FEATURE FUSION AND UPDATE EQUATION

As shown in last section, for a given target entity v, there are three structure-specific embeddings which capture different structural information around the entity. Here we use three corresponding attention values which are learned automatically to distinguish the importance of different structural

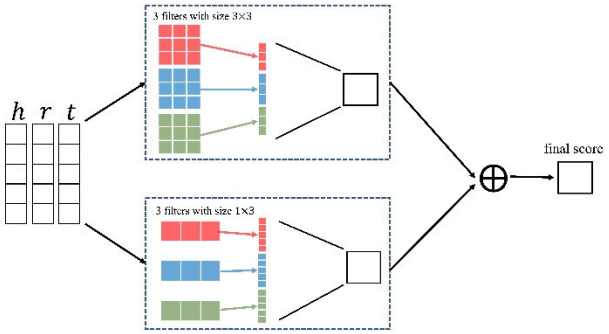


FIGURE 12. The architecture of our decoder. The decoder is based on CNN which uses 3 × 3 filters and 1 × 3 filters to extract global and transitional characteristics for final score of triples.

information and integrate different structural information:

$$h_v^{'} = \alpha_c h_v^c + \alpha_t h_v^t + \alpha_s h_v^s \tag{18}$$

where h_v^c, h_v^t, h_v^s denote 2-cycle-structered, triangle-structured and star-structured embeddings of the target entity respectively, α_c, α_t and α_s are the weights for the corresponding structures, $h_v^{'}$ is the new learned embeddings for the target entity. Moreover, to avoid losing entities' initial information, we perform a linear transformation on initial entity embeddings. And then We add this initial entity embedding information to the entity embeddings obtained from the final attentional layer:

$$H^{''} = W^E H^{'} + H^{init} \tag{19}$$

where H^{init} denotes the initial entity embeddings, $H^{'}$ represents the learned entity embedding from the final attentional layer, W^E is a weight matrix. To get the final relation embeddings, we perform a linear transformation on initial relation embedding matrix R, parameterized by a weight matrix W^R:

$$R^{''} = W^R R \tag{20}$$

D. DECODER

Following ConvE [13] and SACN [17], we use a CNN-based neural network cell as the decoder. Specifically, we use 3×3 convolution filters and 1×3 convolution filters to learn the feature map of triples. The 3×3 convolution filters are aimed to exam the global relationships between different dimensional entries of the embedding triple $[v_h, v_r, v_t]$ for obtaining the interaction of the whole triple while the 1×3 convolution filters are focus on generalizing the transitional characteristics. Figure 12 shows the architecture of our decoder. And the total score function is defined as below:

$$\begin{aligned} f(h,r,t) = &\, concat(\sigma([v_h, v_r, v_t]\Omega_1)) \cdot W_1 \\ &\oplus concat(\sigma([v_h, v_r, v_t]\Omega_2)) \cdot W_2 \end{aligned} \tag{21}$$

where σ is some activation function such as Sigmoid or ReLU; Ω_1 and W_1 are shared parameters in 3×3 convolution layer; Ω_2 and W_2 are shared parameters in 1×3 convolution layer used to compute the score of the triple; denotes the convolution operator; and concat denotes the concatenation operator; $\oplus$ denotes the sum operator to get the final score.

E. TRAINING OBJECTIVE FUNCTION

The encoder and decoder are jointly trained within the end-to-end model. Following the idea of TransE [6] which uses learned embeddings to calculate a triple's score on the condition that $v_h + v_r \approx v_t$, where v_h, v_r, v_t are the embedding of head, relation and tail respectively, we use a scoring function with hinge-loss to train the encoder based on our proposed local structure aware GAT (LSA-GAT). Given a set of valid triples and a set of invalid triples which is created by replacing the head or the tail entity, namely,

$$\begin{aligned} S &= \{(h,r,t) | h,t \in E, r \in R\} \\ S^{'} &= \{(h^{'},r,t) | (h^{'},r,t) \notin S \cup \{(h,r,t^{'}) | (h,r,t^{'}) \notin S \end{aligned}$$

We train the encoder in our model by the following equation, specifically we want the valid and invalid triple to be separated as much as possible:

$$L(\Omega) = \sum_{s \in S} \sum_{s' \in S'} \max(f(s) - f(s') + \gamma, 0) \tag{22}$$

$$f(s) = ||v_h + v_r - v_t||_1, \quad s = (h,r,t) \in S \tag{23}$$

where $\gamma > 0$ is a margin hyper-parameter. And the decoder is trained with the following soft-margin loss function:

$$\begin{aligned} L_{decoder} = &\sum_{(h,r,t) \in S \cup S'} log(1 + exp(y_{(h,r,t)} f(h,r,t))) \\ &+ \frac{\lambda}{2} ||W_1||_2^2 + \frac{\lambda}{2} ||W_2||_2^2 \end{aligned} \tag{24}$$

$$y_{(h,r,t)} = \begin{cases} 1, & (h,r,t) \in S \\ -1, & (h,r,t) \in S' \end{cases} \tag{25}$$

where S' is a collection of invalid triples constructed by replacing the head or tail entity of valid triples in a KG. $f(h,r,t)$ is the score of triple calculated by the decoder.

IV. EXPERIMENTS

A. EXPERIMENTAL SETUP

1) DATASETS

We evaluate the model on two standard benchmark datasets for KG completion, FB15k-237 [21] and WN18RR [13].

FB15k-237: The FB15k-237 dataset contains textual descriptions of knowledge graph relation triples and entity pairs in Freebase. And the dataset is a subset of the FB15k dataset. As mentioned by Toutanova and Chen [21], performing link prediction on FB15k is relatively easy because they contain many reversible relations. This characteristic allows most models to easily predict the majority of test triples. Therefore, in order to accurately test the prediction capability of models, FB15k-237 is constructed by removing the inverse relations in FB15k.

WN18RR: The WN18RR dataset is created from WN18 [6], which is a subset of WordNet. Same as FB15k-237, WN18RR dataset is constructed to ensure that the test data does not have inverse relation test leakage. Each dataset is divided into three sets for training, validation and test. Table 1 shows the statistics of datasets:

TABLE 1. Statistics of the datasets.

Dataset	#Entities	#Relations	#training triples	#validation triples	#test triples
WN18RR	40943	11	86835	3034	3134
FB15k-237	14541	237	272115	17535	20466

2) HYPER-PARAMETERS SETTINGS

We use grid search during the training stage to determine the optimal hyperparameters. We finally determine the hyperparameter ranges: (i) for FB15k-237 dataset, we set 0.001 for learning-rate, 0.03 for dropout, 100 for embedding size, 1 for margin size used in objective function; (ii) for WN18RR dataset, we set 0.0005 for learning rate, 0.01 for dropout, 50 for embedding size, 5 for margin size.

3) EVALUATION PROTOCAL

Following the evaluation protocal of Dettmers [13], each test triple (h, r, t) is converted into two link prediction format: head link prediction $(h, r, ?)$ and tail link prediction $(h, r, ?)$. For every format, the correct entity is ranked among all KG entities excluding the set of other true entities for the triples observed in train and valid sets. In this work, we report the Mean Rank (MR), the Mean Reciprocal Rank (MRR) of the correct entity which is the average of the reciprocal rank of the correct entity, and the Hits@N which is the accuracy in the top N predictions. Here, we choose N as 1, 3 and 10.

B. RESULTS

1) LINK PREDICTION RESULT

Table 2 shows the link prediction results on two datasets. Compared with previous results, our proposed model LSA-GAT achieves better performance on most indicators which shows our model has stronger representation learning ability on knowledge graphs. More specifically, on FB15K-237 dataset, LSA-GAT achieves improvements of 15% in Hits@1, 11% in Hits@3, 7% in Hits@10, and 12% in MRR compared with the second-best results. On WN18RR dataset, LSA-GAT achieves improvements of 4% in Hits@10 and 607 in MR. These results clearly prove that our model significantly outperforms state-of-art results on four metrics for FB15K-237, and on two metrics for WN18RR. We can see that there has a significant improvement on dataset FB15k-237, mainly due to the reason that FB15k-237 dataset has more complex graph structure (more types of relation and less types of entities which mean more complex associations), and then there are more local structures around an entity. Then our model is better at aggregating information within these local structures. The improvement on the dataset WN18RR is not as much as that on dataset FB15k-237. We believe that the reason is that the graph structure of tht dataset WN18RR is relatively similar, so that the local structure we defined is more fixed on the dataset (the relation or path occurred in local structure is fixed). However, our model still achieves competitive performance compared with current mainstream models on dataset WN18RR, especially in Mean Rank (MR), our model achieves the best result with a large gap.

2) COMPARISON STUDY

We conducted a comparison study to evaluate the effect of our aggregation scheme on the model's performance. To quantify the improvements brought by our model, we use the aggregation scheme used by KBAT [19] which belongs to graph attention networks mentioned in section III(B) which ignores local structures and only aggregate one-hop neighboring entities' information. The aggregation scheme in pure graph attention networks is defined as:

$$h^{(l+1)} = \sigma(\sum_{j \in N_i} \alpha_{ij} h_j^l w^l) \tag{26}$$

where α_{ij} is the attention score between entity i and entity j, we compare the performance of two models at different epochs? Figure 13 shows the comparison result of the two models. As the result shown, LSA-GAT (blue) is always better than the comparison model (orange) on FB15K-237 and WN18RR. The result gap between these two models represents the improvement of our model. Changing the way that the neighbor information is aggregated has a large impact on the link prediction results, which shows that our proposed model improves the performance of knowledge base completion.

3) EFFECTIVENESS OF LOCAL STRUCTURE

We conducted an ablation study to evaluate the effect of our defined local structure on the performance of the model. Specifically, we remove any one of the three structures and keep the other two to compare the experimental results, where we analyze the convergence speed, Hits@1 and mean rank on the reduction model and the original model. Figure 14 shows the experimental result.

As is shown in the experimental results, when we remove one of the three structures, the convergence speed of the encoder (the neighbor aggregation function) is lower than that of the model without any reduction, and the result of link prediction is slightly lower than the original model. The fact that the blue line overlaps the gray line shows the convergence speed of the model which removes the triangle structure is similar to the model which removes the star structure.

From the experimental results, we can see that the effect drops most obviously when the start-structure is removed, which shows that our model learns different semantic from other entities by fusing the information of different entities under different relations as much as possible. Meanwhile, the experimental results show that removing the triangular structure has more obvious effects than removing the cyclic structure, which indicates that the simultaneous integration of path information and relationship information around the entity has an important effect. Finally, we notice from figure 13(a) that the convergence speed of our model is the fastest which means that the model we designed can better learn entity representations which contains richer semantic information to help the knowledge graph completion tasks. This shows that the local structure defined in this paper can improve the performance of link prediction.

TABLE 2. Experimental results including Hits@N, MRR and MR on FB15K-237 and WN18RR test sets. The best score is in bold and second-best score is underlined.

Dataset	FB15k-237					WN18RR				
	Hits@N			MRR	MR	Hits@N			MRR	MR
	1	3	10			1	3	10		
Distmult(Yang et al., 2014)	0.16	0.26	0.42	0.24	254	0.39	0.44	0.49	0.43	5110
ComplEx (Trouillon et al., 2016)	0.16	0.28	0.43	0.25	339	0.41	0.46	0.51	0.44	5261
ConvE (Dettmers et al. 2017)	0.23	0.35	0.50	0.32	244	0.40	0.44	0.52	0.43	4187
ConvKB (Nguyen et al., 2018)	0.20	0.32	0.52	0.40	257	0.06	0.45	0.53	0.25	2554
R-GCN (Schlichtkrull et al., 2018)	0.15	0.26	0.42	0.25	--	--	--	--	--	--
SACN (Shang et al., 2019)	0.27	0.40	0.55	0.36	--	0.43	0.48	0.54	0.47	--
CompGCN(Shikhar Vashishth et al.,2020)	0.26	0.39	0.53	0.35	**197**	**0.44**	0.49	0.54	**0.47**	3533
LSA-GAT(our proposed method)	**0.41**	**0.50**	**0.60**	**0.47**	273	0.35	**0.49**	**0.58**	0.44	**1947**

TABLE 3. Number of local structures analysis on FB15k-237 Dataset. We perform both horizontal and vertical comparison to confirm the effectiveness of our defined structures.

Dataset	#local structures/#degree	Hits@1	Hits@3	Hits@10	MRR	MR
GAT	10-50	0.15	0.23	0.36	0.21	552
	over 50	0.46	0.57	0.67	0.54	327
LSA-GAT	10-50	0.15(−)	0.24(↑)	0.37(↑)	0.22(↑)	427(↑)
	over 50	0.51(↑)	0.61(↑)	0.69(↑)	0.58(↑)	220(↑)

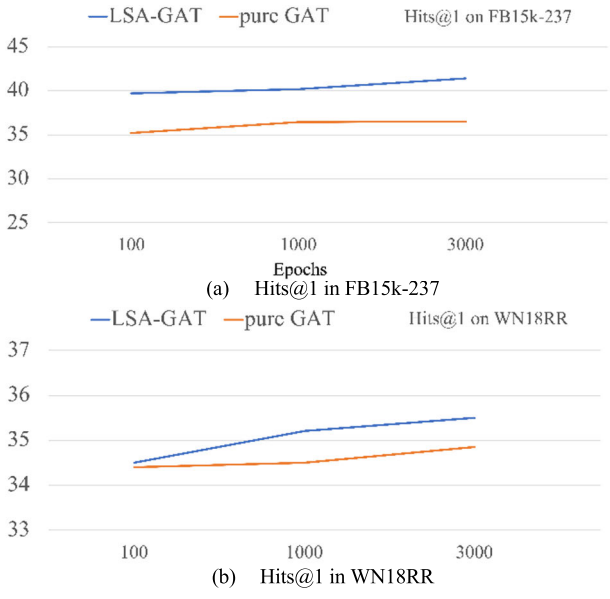


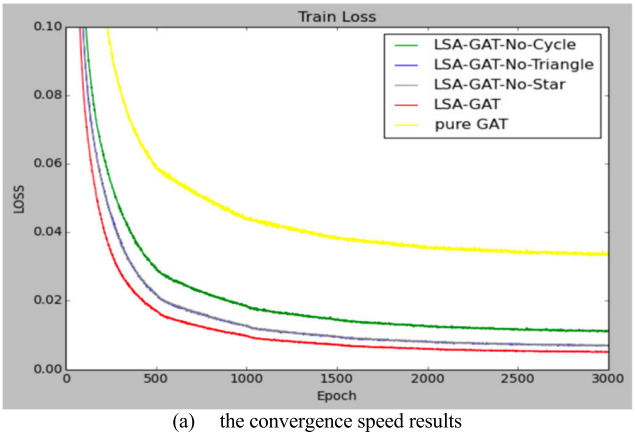
FIGURE 13. The comparison study of LSA-GAT in WN18RR and FB15K-237.

4) NUMBER OF LOCAL STRUCTURE ANALYSIS

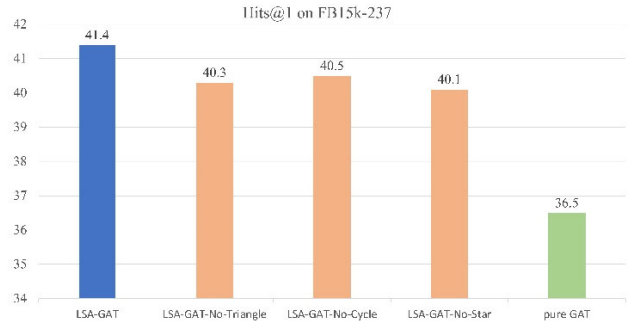
The number of local structures can reflect the environmental characteristics of the target entity. An entity has more opportunities to achieve special semantic interactions with other entities when it is surrounded by more diverse structures. As a comparison, we studied the influence of node degree in the pure GAT model, as higher degrees indicate diverse local structures. We use the same magnitude of degree and local structures on these two models to compare their performance. As shown in Table 3, we present the results for different sets of nodes with different degree scopes and different local structures scopes for the pure GAT model and the LSA-GAT model respectively.

According to the experimental results, we illustrate the effectiveness of our model from two perspectives. First, in terms of horizontal comparison, when there are more local special structures around a target entity, our model has better performance in predicting the facts about the entity, Hits@1 changed from 0.15 to 0.51, Hits@3 from 0.24 to 0.69 (shown in yellow box). Considering that more neighbors could reveal more local structures, we designed a vertical comparison, under the conditions of **the same magnitude of the number of degree and local structures**, our proposed model LSA-GAT is better than pure GAT, Hits@1 changed from 0.46 to 0.51, Hits@3 from 0.57 to 0.61 (showed in red box). This shows that compared to the pure GAT model that uses one-hop direct neighbor information, our proposed model LSA-GAT learns richer semantic information by fusing different special local structures, which significantly improves the performance of link prediction.

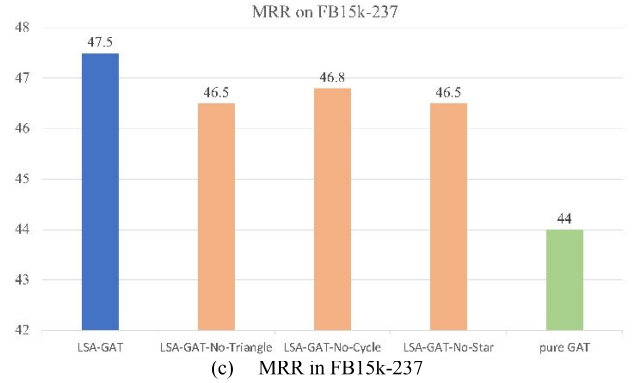
To further illustrate the effectiveness of our model, we extract a subgraph around an entity and learn the information of neighboring local structures defined in this work, then



(a) the convergence speed results



(b) Hits@1 in FB15k-237



(c) MRR in FB15k-237

FIGURE 14. The ablation study results of LSA-GAT on FB15K-237.

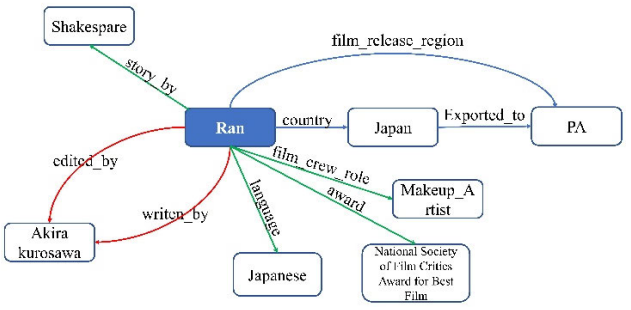


FIGURE 15. The subgraph of movie entity "Ran".

we complete missing facts about the entity using our model. Figure 15 shows a subgraph around the movie entity *Ran*. The blue, red, green links are the previously-designed triangle structure, 2-cycle structure, and star structure respectively.

For the question *(Oscar_1985, hornored_for,* ?) and *(?,award, London_Film_Critics_Circle_Award_for_Best _Director)*, our model ranks the answer entity ***Ran*** within 1 and 3 respectively. But meanwhile the pure GAT model rank both over 10. This shows our model has learned a more appropriate embedding for the entity ***Ran*** and better infer missing triples about ***Ran***.

V. CONCLUSION AND FUTURE WORK

In this paper, we propose a local structure aware model called LSA-GAT for knowledge graph completion. Three typical local structures are defined to capture the semantic connections between entities and relations, and the information beneath these structures is aggregated to learn the embeddings of entities and relations. Our work follows an encoder-decoder fashion. The encoder is focused on embedding entities and relations, and the decoder applies 3×3 convolution filters and 1×3 convolution filters which capture the global interaction and translational characteristic between entities and relations respectively to score a would-be fact. Compared with the current state-of-the-art model, our proposed model LSA-GAT has a significantly improvement.

In the future, we intend to extend our method to better perform on more complex structures such as sub-graph and capture higher-order structures between entities (within multi-hops) in our graph attention model. Another intention is trying to incorporate attention-based walking into our encoder to focus on small but informative structures of the graph.

REFERENCES

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

[2] F. M. Suchanek and G. Kasneci, "Yago: A core of semantic knowledge," in *Proc. 16th Int. Conf. World Wide Web (WWW)*, 2007, pp. 697–706.

[3] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and I. Zachary, "Dbpedia: A nucleus for a Web of open data," in *The semantic Web*. Berlin, Germany: Springer, 2007, pp. 722–735.

[4] A. Carlson, J. Betteridge, B. Kisiel, and B. Settles, "Toward an architecture for never-ending language learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 5, no. 3. Atlanta, GA, USA, Jul. 2010,

[5] O. Etzioni, M. Banko, S. Soderland, and D. S. Weld, "Open information extraction from the Web," *Commun. ACM*, vol. 51, no. 12, pp. 68–74, 2008.

[6] 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.

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

[8] Y. Lin, "Learning entity and relation embedding for knowledge graph completion," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2181–2187.

[9] 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.

[10] B. Yang, W. T. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," Aug. 2015, *arXiv:1412.6575*. [Online]. Available: https://arxiv.org/pdf/1412.6575

[11] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *Proc. ICML*, 2016, pp. 2071–2080.

[12] I. Balazevic, C. Allen, and T. M. Hospedales, "TuckER: Tensor factorization for knowledge graph completion," in *Proc. EMNLP*, 2019, pp. 5188–5197.
[13] T. M. S. Dettmers Pasquale Pontus and S. Riedel, "Convolutional 2D knowledge graph embeddings," in *Proc. 32th AAAI Conf. Artif. Intell.*, 2018, pp. 1811–1818.
[14] 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, pp. 1–10.
[15] S. Vashishth, "InteractE: Improving convolution-based knowledge graph embeddings by increasing feature interactions," in *Proc. 32th AAAI Conf. Artif. Intell.*, 2020, pp. 3009–3016.
[16] 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.
[17] C. Shang, Y. Tang, J. Huang, J. Bi, X. He, and B. Zhou, "End-to-end structure-aware convolutional networks for knowledge base completion," in *Proc. 33th AAAI Conf. Artif. Intell.*, 2019, pp. 3060–3067.
[18] S. Vashishth, S. Sanyal, V. Nitin, and P. Talukdar, "Composition-based multi-relational graph convolutional networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2020.
[19] D. Nathani, J. Chauhan, C. Sharma, and M. Kaul, "Learning attention-based embeddings for relation prediction in knowledge graphs," 2019, *arXiv:1906.01195*. [Online]. Available: http://arxiv.org/abs/1906.01195
[20] J. B. Lee, R. Rossi, and X. Kong, "Graph classification using structural attention," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1666–1674.
[21] 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.
[22] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" 2018, *arXiv:1810.00826*. [Online]. Available: http://arxiv.org/abs/1810.00826
[23] K. Li, G. Lu, G. Luo, and Z. Cai, "Seed-free graph de-anonymiztiation with adversarial learning," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2020, pp. 745–754.
[24] J. Chen, C. Wang, H. Lin, W. Wang, Z. Cai, and J. Wang, "Learning the structures of online asynchronous conversations," in *Proc. 22nd Int. Conf. Database Syst. Adv. Appl. (DASFAA)*, 2017, pp. 19–34 .
[25] K. Zhang, Y. Zhu, J. Wang, and J. Zhang, "Adaptive structural fingerprints for graph attention networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019.
[26] J. Jiang, S. Wen, S. Yu, Y. Xiang, and W. Zhou, "Identifying propagation sources in networks: State-of-the-Art and comparative studies," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 1, pp. 465–481, 1st Quart., 2017, doi: 10.1109/COMST.2016.2615098.
[27] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*. [Online]. Available: http://arxiv.org/abs/1710.10903
[28] Z. Xiong, W. Li, Q. Han, and Z. Cai, "Privacy-preserving auto-driving: A GAN-based approach to protect vehicular camera data," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2019, pp. 668–677.

KEXI JI was born in Dazhou, Sichuan, China, in 1997. She received the B.E. degree from the Department of Computer Science and Engineering, Sichuan Normal University, China, in 2019. She is currently pursuing the master's degree with the College of Information and Software Engineering, University of Electronic in Science and Technology of China.

Her main research interest includes the construction of knowledge graphs, including entity extraction, entity disambiguation, and knowledge reasoning.

BEI HUI received the Ph.D. degree from the University of Electronic Science and Technology of China (UESTC), in 2009. He is currently an Associate Professor with the School of Information and Software Engineering, University of Electronic Science and Technology of China. His research interests include machine learning and medical image processing.

GUANGCHUN LUO (Member, IEEE) received the M.S. and Ph.D. degrees from the University Electronic Science and Technology of China (UESTC), in 1999 and 2004, respectively.

He joined the University of California at San Diego and the University of Toronto, Canada, as a Visiting Scholar. He is currently a Professor and the Vice President of the Graduate School, UESTC. He has published over 50 papers in international journals and conference proceedings. His research interests include big data technology and its application, cloud computing, new network technology, middleware technology, and network security.

Dr. Luo has also been very active and serves for a number of technical societies, including as a Committee Member of the Calculation of Sichuan Province Institute of High Performance Computer and China Education Information Council. He has received regular funding from various departments of the Chinese Government, such as the National Natural Science Foundation of China, the National Hi-Tech Research and Development Program (the 863 Program), and the Science and Technology Department Foundation of Sichuan Province.