Neural Computing and Applications (2020) 32:14963–14973
https://doi.org/10.1007/s00521-020-04851-5

ORIGINAL ARTICLE

KGAnet: a knowledge graph attention network for enhancing natural language inference

Meina Song[1] · Wen Zhao[1] · E. HaiHong[1]

Received: 15 May 2019 / Accepted: 14 March 2020 / Published online: 26 June 2020

Abstract
Natural language inference (NLI) is the basic task of many applications such as question answering and paraphrase recognition. Existing methods have solved the key issue of how the NLI model can benefit from external knowledge. Inspired by this, we attempt to further explore the following two problems: (1) how to make better use of external knowledge when the total amount of such knowledge is constant and (2) how to bring external knowledge to the NLI model more conveniently in the application scenario. In this paper, we propose a novel joint training framework that consists of a modified graph attention network, called the knowledge graph attention network, and an NLI model. We demonstrate that the proposed method outperforms the existing method which introduces external knowledge, and we improve the performance of multiple NLI models without additional external knowledge.

Keywords Natural language processing · Natural language inference · External knowledge

1 Introduction

Natural language inference (NLI), also known as recognizing textual entailment, is a challenging and fundamental task in natural language understanding. Its aim is to determine the relationship (entailment, neutral, or contradiction) between a premise and hypothesis.

In the past few years, large annotation datasets, such as the Stanford NLI (SNLI) dataset [1][1] and the Multi-Genre NLI (MultiNLI) corpus [2],[2] have been provided, which has made it possible to train quite complex neural network-based models that are suitable for a large number of parameters to better solve NLI problems. These models are divided into two main categories, sentence-encoding and inter-sentence models.

Sentence-encoding models use the Siamese structure [3] as a reference, for encoding premises and hypotheses into sentence vectors and then for comparing the distance of the sentence vectors to obtain the relationship categories. Talman et al. [4] used hierarchical biLSTM and a max pooling architecture to encode sentences into vectors. Nie et al. [5] used shortcut-stacked sentence encoders to perform multi-domain semantic matching. Shen et al. [6] applied a hybrid of hard and soft attention and reinforcement learning for modeling sequences. Im and Cho [7] proposed a distance-based self-attention network, which considers the word distance using a simple distance mask. Yoon et al. [8] designed dynamic self-attention by modifying the dynamic routing in a capsule network [9] for natural language processing. Encoders include convolutional neural networks (CNNs) [10], recurrent neural network variants [1, 5, 11], and self-attention networks [12].

In contrast to the above methods, inter-sentence models apply cross-attention to increase the interaction between sentences. Among them, Parikh et al. utilized a decomposition matrix to reduce the number of cross-attention parameters in their proposed model, called DecAtt [13]. Gong et al. [14] introduced the interactive inference network, which is able to achieve a high-level understanding of a sentence pair by hierarchically extracting semantic

✉ E. HaiHong
ehaihong@bupt.edu.cn

Wen Zhao
luzhizhaowen@bupt.edu.cn

[1] Beijing University of Posts and Telecommunications, Beijing, China

[1] https://nlp.stanford.edu/projects/snli/.

[2] https://www.nyu.edu/projects/bowman/multinli/.

features from the interaction space. Tan et al. [15] designed four attention functions to match words in corresponding sentences. Chen et al. proposed an enhanced sequence-encoding model (ESIM) [16]. In addition, Wang et al. obtained inter-sentence relationship features from multiple perspectives (BiMPM) [17]. Finally, Kim et al. [18] proposed a densely connected convolutional network that enables the original and the co-attentive feature information from the bottommost word embedding layer to be preserved in the uppermost recurrent layer.

Using inter-sentence relationships to handle the occurrence of unknown relationships in the corpus, Chen et al. proposed the knowledge-based inference model (KIM) [20], which uses the word-pair relationship features extracted from WordNet [19]. In their work, the model consists of knowledge-enriched co-attention, local inference collection with external knowledge, and knowledge-enhanced inference composition. Specifically, taking Fig. 1 as an example, after extracting the triple (round, synonymy, circular) from WordNet, the synonymy representation is added to the three components. Meanwhile, they obtained a certain improvement by establishing a Co-hyponyms relationship for two entities that have the same hypernym but do not belong to the same synset in the WordNet. The effectiveness of KIM verifies the validity of external knowledge. However, the two following problems exist: First, compared to subgraphs containing entities and relationships, triples in the knowledge graph are just a very simple structure. There is still great potential on the way to distill the knowledge graph (sufficiency). Second, this method cannot be used directly in other models. In the field of NLI, we need a general approach to introducing knowledge, without dealing with graph data and heavily modifying the internal structure of the model (applicability).

Inspired by KIM [20] and the use of graph networks to extract knowledge graphs in other fields (elaborated in related work), we propose a framework to provide external knowledge to NLI models, that is, to integrate the information of the whole subgraph using a graph attention network (GAT) [21], and to train the GAT and NLI model jointly.

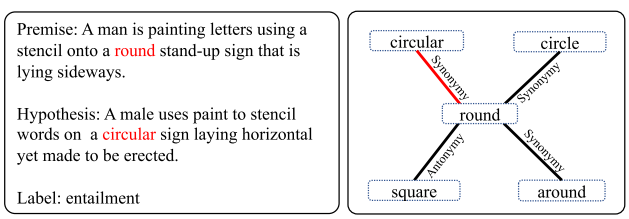


Fig. 1 Example from the SNLI [1] dataset. The right graph is a subgraph extracted from WordNet [19]. Words in red are entities appearing in the subgraph, and the edge in red indicates the relationship between them (color figure online)

Moreover, the GAT is applied to homogeneous networks such as citation networks [22]. To make the GAT more suitable for extracting information from knowledge graphs, we add a relationship importance attribute to the GAT and call it a knowledge GAT (KGAnet).

In experiments, we demonstrate that KGAnet outperforms the previous work and improves the performance of multiple NLI models without external knowledge. Finally, we verify the correctness of adding relationship attributes.

2 Related work

2.1 Graph convolutional networks

A CNN [23] is a neural network with powerful functionality in computer vision and natural language processing, because it can extract the spatial features of Euclidean-structured data. However, many data are non-Euclidean in structure, such as social networks and molecular structures. A graph convolutional network is used to process such data. The function of the graph convolutional network is to fuse the nodes and the surrounding information to obtain richer and more accurate representations of the nodes. In recent years, many researchers have conducted studies in the field of graph convolution, and its methods fall into two main categories: spectral methods and non-spectral methods.

Non-spectral methods The non-spectral methods [24] directly define the convolution on a graph using an adjacency matrix to summarize the node features of all spatial neighboring matrices. The main challenge of non-spectral methods comes from the dynamic size of the neighborhood. Duvenaud et al. [25] proposed a CNN that runs directly on the original molecular graphs and learns a specific weighted matrix for each node. To enable traditional CNNs to be directly used with graph input, they select fixed-size neighbors and normalize these nodes [26]. Recently, Hamilton et al. [27] introduced the task of inductive node classification with the aim of classifying nodes that have not been seen during training. This method takes a fixed-size neighborhood of the node and then uses a specific aggregator (such as the mean operator or LSTM [28]) to learn the neighborhood features. It has achieved impressive results on several large-scale inductive benchmark tests.

Spectral methods Bruna et al. [29] first proposed the definition of a graph convolution network in the Fourier domain. Because the convolution definition contains a Laplacian feature decomposition of the graph, the calculation process is quite arduous. Subsequently, a smooth parametric spectral filter [30], Chebyshev polynomial [31],

and Cayley polynomial [32] were introduced to improve the computational efficiency. Then, Kipf and Welling [33] greatly simplified the convolution operation using a first-order Chebyshev polynomial and achieved good results on node classification. Although the above methods significantly improve the computational efficiency, they depend on the structure of the graph itself and cannot be directly used for graphs of different structures. A GAT [21] introduces an attention mechanism based on graph convolutional networks and calculates the weights of different nodes in the central node domain according to node similarity, which allows the model to accept inputs of different sizes.

However, in a knowledge graph, to enrich the entity expressions, it is obviously not enough to consider the similarity of entities. In contrast to general graph structure data, the more important a relationship is to the central entity, the more important the connected neighbor entities are. It is reasonable for the property of the relationship to participate in the calculation of weights between entities.

2.2 Using graph convolutional networks to introduce external knowledge for other tasks

Story generation In the field of story generation, Jian et al. [34] aligned the entities that appear in the story and ConceptNet [35] and used Graph Attention to generate entity representations that enhanced the model's understanding of the entire story.

Dialogue system Hao et al. proposed a CCM [36] (commonsense knowledge-aware conversational model). Given a user post, the model retrieves the relevant knowledge graph from ConceptNet and then uses the static graph attention mechanism to increase the semantic information of the post. Then, during the word generation process, the model reads the retrieved knowledge graph and produces better generation through the dynamic graph attention mechanism.

CommonsenseQA Bill et al. proposed a knowledge-aware graph network module named KagNet [37], a text reasoning framework for commonsense answering questions, which effectively utilizes ConceptNet to perform explainable reasoning.

Recommendation Xiong et al. termed the hybrid structure of knowledge graph and user–item graph as collaborative knowledge graph (CKG). They developed a knowledge graph attention network (KGAT) [38], which achieved high-order relation modeling in an explicit and end-to-end manner under the graph neural network.

One common feature of their works is they all used graph neural networks with edge information to inject external knowledge for their professional fields. Inspired by these works, in this paper, we try to use a similar structure to design a general way to introduce external knowledge for the NLI domain.

2.3 Using external knowledge to enhance natural language inference

In the NLI field, a lot of works have been devoted to studying the effectiveness of external knowledge and how to introduce external knowledge.

KIM [20] was the first NLI model to introduce external knowledge. Chen et al. [20] enhanced NLI models with external knowledge in co-attention, local inference collection, and inference composition components. They first explored how to introduce external knowledge and proved the effectiveness of external knowledge in the NLI field.

Wang et al. [39] proposed a ConSeqNet system combining text and graph models: This system uses ConceptNet as an external knowledge source to solve natural language inference (NLI) problems. At the same time, they also compared the effects of three external knowledge (wordnet, DBPedia, and ConceptNet) and tried on the diversity of external knowledge.

Annervaz et al. [40] used attention to automatically align entities and words and used convolutional neural network-based methods to extract external knowledge to reduce the scale of the attention space. They demonstrated that model training with the help of external knowledge can converge with fewer labeled samples.

These works of introducing external knowledge in NLI practice were performing well. Unlike them, our work verifies the effectiveness and flexibility of graph convolutional networks in the NLI field.

3 KGAnet framework

In this section, we define a KGAnet and introduce the combination of KGAnet and NLI models.

3.1 KGAnet

For each entity, our aim is to enable the NLI model to fully access external knowledge, so we use a KGAnet to generate representation vectors made up of the entity's neighbors and relationships. In the knowledge graph, we call the first-order neighbor entity j of an entity i a neighbor entity and we call i the core entity. When processing a knowledge graph, one layer of the KGAnet combines the information of the first-order subgraph into the

representation of the core entity. Furthermore, k layer of a KGAnet combines the input features of the k-order subgraph into the representation. In this study, we only consider first-order neighbors; the effect of the k-layer neighbors will be considered in future work. The KGAnet inputs are a set of entity features $h = \{\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_N\}$ and a set of relationship features $r = \{\boldsymbol{r}_{11}, \boldsymbol{r}_{12}, \ldots, \boldsymbol{r}_{NN}\}$. Specifically, $\boldsymbol{h}_i \in \mathbb{R}^F$ is a representation of entity i, and $\boldsymbol{r}_{ij} \in \mathbb{R}^F$ is the relationship between entities i and j. The output is a new set of entity features $h^{'} = \{\boldsymbol{h}_1^{'}, \boldsymbol{h}_2^{'}, \ldots, \boldsymbol{h}_N^{'}\}$, $\boldsymbol{h}_i^{'} \in \mathbb{R}^{F^{'}}$, where N is the number of entities of graph, F is the number of dimensions of the input entity features, and we denote the output node feature dimension as $F^{'}$.

First, we calculate the degree of importance of the neighboring entity with respect to the core entity. The weight calculation involves two parts: the importance of the relationships and the similarity between entities. We, respectively, define a relationship's importance and entities similarity as follows:

$$I_{ij} = W_r \boldsymbol{r}_{ij}, \tag{1}$$

$$M_i = W_h \boldsymbol{h}_i, \tag{2}$$

$$M_j = W_h \boldsymbol{h}_j, \tag{3}$$

$$S_{ij} = \tanh(W_a M_i + W_a M_j). \tag{4}$$

In these formulas, $W_r \in \mathbb{R}^F$, $W_h \in \mathbb{R}^{F^{'} \times F}$ and $W_a \in \mathbb{R}^{F^{'}}$ are linear transformations. We use W_h to filter entity features, W_r to filter relationship features, and W_a to calculate the correlation of entity features. We use I_{ij} to denote a score function that determines the importance of the relationship $\boldsymbol{r}_{ij}$ and use S_{ij} to denote a score function for the similarity between core entity i and entity j. When both I_{ij} and S_{ij} are small, we want to get a small value of E_{ij}. When I_{ij} and S_{ij} are both large, we want the value of E_{ij} to be large. So we multiply the two scores to get the weights E_{ij} between the entities as follows:

$$E_{ij} = I_{ij} \times S_{ij}. \tag{5}$$

There is no I involved in the calculation in GAT [21]. The entity i and its neighbor entity j form a subgraph. Entity i and entity j in E_{ij} may not be associated with each other. We need the adjacency matrix A of the graph to act as a mask, erasing the relationships that do not exist in E_{ij}. The specific operation is:

$$E_{ij} = \begin{cases} E_{ij} & A_{ij} = 1 \\ -\infty & A_{ij} = 0 \end{cases}. \tag{6}$$

A_{ij} is the adjacency matrix of the graph. If there is an relationship between entity i and entity j, the value of A_{ij} is 1; otherwise, the value is 0. When A_{ij} is 0, the value of E_{ij} is negative infinity to obtain a value close to 0 after normalization.

To make the weights of all neighbor entities of i easy to compare, we normalize E_{ij} as follows:

$$\alpha_{ij} = \mathrm{softmax}_j(E_{ij}) = \frac{\exp(E_{ij})}{\sum_{k \in N_i} \exp(E_{ik})}. \tag{7}$$

After α_{ij} obtained, we use it to sum over all neighbor entities according to importance as follows:

$$\boldsymbol{h}_i^{'} = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} M_j\right), \tag{8}$$

where $\mathcal{N}_i$ is some neighborhood of node i in the graph and M_j is the neighbor feature after filtering. The new features $\boldsymbol{h}_i^{'}$ of entity i fuse the information about its surrounding entities and relationships. σ is a sigmoid function. During the training process, parameters W_r, W_a, and W_h are all trained: W_r and W_a learn how to provide a reasonable score for the relationship importance and entities similarity and W_h acquires the ability to select beneficial features from neighbor entities.

3.2 Basic components of NLI models

At present, for inter-sentence methods, cross-attention is widely used in NLI models and is regarded as the basic component of such models. Note that cross-attention can align words that are related between sentences. If the input word vector contains an association, then this association feature will be mined in the cross-attention. Below, we will briefly introduce the basic framework of all current inter-sentence models and mainly describe what the cross-attention is and its function.

Figure 2 shows a typical inter-sentence model structure. The premise $a = \{{w_1}^a, {w_2}^a, \ldots, {w_{l_a}}^a\}$ and hypothesis $b = \{{w_1}^b, {w_2}^b, \ldots, {w_{l_b}}^b\}$ are the input of the model. We denote the ith word in the premise as ${w_i}^a$ and the jth word in the hypothesis as ${w_j}^b$.

Cross-attention finds the degree of association of each word in two sentences. Using the degree of association, each sentence uses the other sentence's word information to enhance its own word representation, completing a soft alignment of the two sentences. In this part, the relationships of the entities are extracted to enhance the interaction of the two sentences. In the cross-attention layer, we can calculate the similarity of words between sentences using the following equation:

$$\beta_{ij} = {w_i}^{aT} {w_j}^b. \tag{9}$$

Then, we need to align the similar words in the two sentences as follows:

$$\tilde{a}_i = \sum_{j=1}^{l_h} \frac{\exp(\beta_{ij})}{\sum_{k=1}^{l_h} \exp(\beta_{ik})}, \quad \forall i \in [1, 2, \ldots, l_p], \tag{10}$$

$$\tilde{b_j} = \sum_{i=1}^{l_p} \frac{\exp(\beta_{ij})}{\sum_{k=1}^{l_p} \exp(\beta_{kj})}, \quad \forall j \in [1, 2, \ldots, l_h]. \tag{11}$$

Next, we can perform down-sampling on each sentence through max pooling and average pooling to obtain vectors V_p and V_h as follows:

$$V_{a,\text{ave}} = \sum_{i=1}^{l_a} \frac{\tilde{a}_i}{l_a}, V_{a,\max} = \max_{i=1}^{l_a} \tilde{a}_i, \tag{12}$$

$$V_{b,\text{ave}} = \sum_{j=1}^{l_b} \frac{\tilde{b_j}}{l_b}, V_{b,\max} = \max_{j=1}^{l_b} \tilde{b_j}, \tag{13}$$

$$V_a = [V_{a,\text{ave}}; V_{a,\max}], \tag{14}$$

$$V_b = [V_{b,\text{ave}}; V_{b,\max}], \tag{15}$$

$$V = [V_a - V_b; V_a \odot V_b; V_a; V_b]. \tag{16}$$

We concatenate V_a, V_b, their difference, and their element-wise product into a vector V, as shown in Eq. 16. Finally, we feed V into a multilayer perceptron classifier (MLP). MLP includes two layers of feedforward neural network, which have *Hidden_Size* and 3 neurons, respectively. The activation function of the last layer is softmax, and the output is the relationship between the two sentences.

3.3 KGAnet and NLI models

To obtain external knowledge, it is convenient to concatenate the output of KGAnet and the input of NLI and jointly train them. The combined method is shown in Fig. 3. The vector $\boldsymbol{Q_k} \in \mathbb{R}^Q$ of each word is composed of a pre-trained D-dimensional word vector $\boldsymbol{w_k}$ and a new entity feature $\boldsymbol{h_i'}$ obtained by KGAnet, as shown in Eq. 17, where W_q is a linear transformation.

$$\boldsymbol{Q_k} = W_q[\boldsymbol{h_i'}; \boldsymbol{w_k}]. \tag{17}$$

Considering that not every word will find related entities in the graph, so some words in the sentence will correspond to empty entities. For these entities, we assign the following values:

$$\boldsymbol{h_\phi} = \frac{\sum_{i=1}^{N} \boldsymbol{h_i}}{N}. \tag{18}$$

After $\boldsymbol{Q_k}$ is obtained, it is used as the new input data for the NLI model and is passed to various encoders. For instance, in DecAtt [13], encoder is a fully connected layer. In ESIM [16] and BiMPM [17], encoder is bidirectional LSTM [28].

4 Experiment

4.1 Experimental setup

In the experiment, all of our models use the Adam optimizer [41]. The word vector is a pre-trained GloVe [42][3] 300-dimensional vector. For words that are out-of-vocabulary, we assign them a 300-dimensional zero vector. Table 1 shows the hyper-parameters used for all experiments.

4.2 Effect of hyperparameters

We design an experiment with three hyperparameters (*Batch_Size*, *Hidden_Size*, *Learing_Rate*) of ESIM+KGAnet. The experimental results are shown in Fig 7 in "Appendix." *Hidden_Size* is the number of neurons in the first layer of the multilayer perceptron classifier (MLP) in Sect. 3.2. The experimental results show that: (1) the update step is too large when *Learing_Rate* is greater than 0.001, so our model fail to converge; (2) when *Batch_Size* is less than 32, the optimizer's inaccurate estimate of the gradient results in poor model performance; and (3) *Hidden_Size* has little effect on results. Generally speaking, when the learning rate is not large and the batch size is not small, the robustness of the model can be guaranteed.

4.3 Dataset

In the experiment, the performances of all models were evaluated on the SNLI dataset [1] and the MultiNLI corpus [2]. In the SNLI dataset, there are three possible relationships between a premise and hypothesis: contradiction, neutral, and entailment. This dataset consists of a 549,367-sample training set, 9842-sample development set, and a 9842-sample test set. The MultiNLI dataset consists of a 392,703-sample training set. The test and development sets are split into 10,001-sample in-domain (matched) and 10,001-sample cross-domain (mismatched) sets.

To closely retain the scale of the knowledge graph used in the previous work [20], we chose WordNet [19] to provide the knowledge graph data, and we preprocessed the knowledge graph as follows: (1) we selected 14,216 entities including entities in wordnet that appear in the snli and mnli training sets and their neighbors. (2) We selected six relationships in WordNet: entailment, member meronymy, synonymy, antonymy, hypernymy, and similarity. TransE [43] is a graph embedding technology that we used to embed the 14,216 entities and their relationships as 300-dimensional vectors.

[3] https://nlp.stanford.edu/projects/glove/.

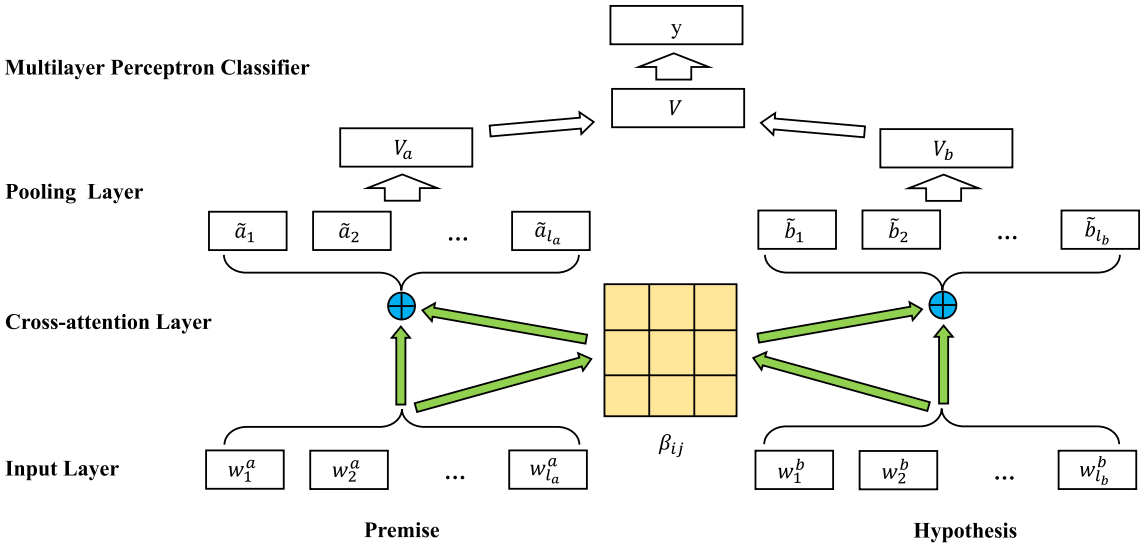


Fig. 2 High-level view of our base model

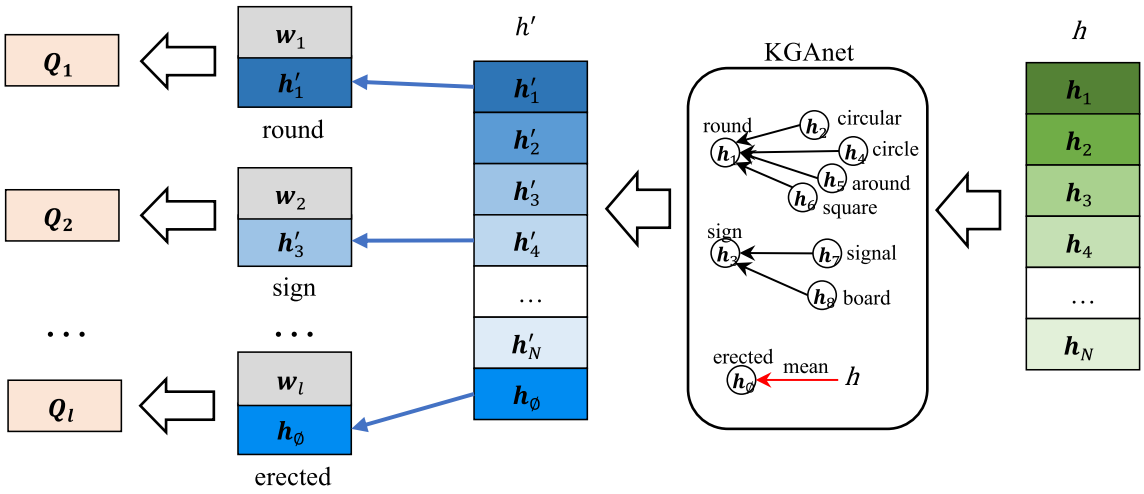


Fig. 3 Method of combining KGAnet with an NLI model. Here, $\boldsymbol{h}$ denotes the KGAnet input data and $\boldsymbol{h}'$ denotes the output of KGAnet. An empty entity is denoted as $\boldsymbol{h}_\phi$. The red arrow indicates that "erected" has no adjacent words in the graph, so take the mean of input h (color figure online)

Table 1 Hyperparameters used in the experiments

Hyperparameter	Value
Learning rate	0.001
Word vector dimension	300
Hidden-state dimension	300
F	300
F'	300
Batch size	64
Epochs	10

F, the length of the entity and relationship vectors; F', the length of the KGAnet output entity vector

4.4 Performance of KGAnet

In view of the problems we attempt to address in Sect. 1, we evaluated two aspects of our approach: sufficiency and applicability. The sufficiency is demonstrated by comparing the testing accuracies and training processes of KGAnet and KIM [20], which is the previous method for acquiring external knowledge. The applicability is reflected by the improvements obtained by adding KGAnet to other NLI models without external knowledge.

Table 2 Test accuracy improvement in NLI models after adding KGAnet on the SNLI dataset

Previous model	Baseline accuracy			KGAnet + Baseline accuracy		
	Train	Dev	Test	Train	Dev	Test
DecAtt [13]	89.23 ± 0.25	86.34 ± 0.14	86.3 ± 0.05	89.45 ± 0.24	87.15 ± 0.22	86.93 ± 0.18
BiMPM [17]	89.32 ± 0.07	87.41 ± 0.47	87.53 ± 0.25	90.34 ± 0.05	88.34 ± 0.07	88.12 ± 0.26
KIM [20]	92.31 ± 0.03	88.93 ± 0.08	88.64 ± 0.05	–	–	–
ESIM [16]	91.23 ± 0.15	88.79 ± 0.21	88.02 ± 0.08	92.48 ± 0.17	89.26 ± 0.23	88.92 ± 0.04

Table 3 Test accuracy improvement in NLI models after adding KGAnet on the MultiNLI dataset

Previous model	Baseline accuracy			KGAnet + Baseline accuracy		
	Train	Matched	Mismatched	Train	Matched	Mismatched
DecAtt [13]	77.88 ± 0.07	71.90 ± 0.03	71.34 ± 0.03	78.29 ± 0.05	73.21 ± 0.03	72.66 ± 0.03
BiMPM [17]	78.12 ± 0.08	73.10 ± 0.03	72.46 ± 0.07	78.6 ± 0.04	73.6 ± 0.03	73.41 ± 0.03
KIM [20]	80.71 ± 0.06	74.08 ± 0.05	74.10 ± 0.02	–	–	–
ESIM [16]	81.40 ± 0.04	75.7 ± 0.06	73.83 ± 0.03	82.15 ± 0.15	76.03 ± 0.07	74.22 ± 0.02

4.4.1 Sufficiency

KIM [20], which enriches ESIM [16] with external knowledge, is the first and powerful model using a knowledge graph in the NLI field, so we use it as our comparison model for KGAnet, which we combine with ESIM. Table 2 shows that, on the SNLI dataset, the joint framework of KGAnet and ESIM outperforms KIM by 0.28% with respect to test accuracy. In Table 3, we can see that, on the MultiNLI dataset, the joint framework of KGAnet and ESIM outperforms KIM by 0.1% with respect to accuracy on the mismatch dataset.

Figure 4 displays the training performance of KIM and our model (ESIM + KGAnet) on the SNLI dataset. Our model shows a strong generalization capability early in the training period. Although both models start to converge at nearly the same time, our model is much more accurate than KIM on the development set during epochs 0–3.

Similarly, Fig. 5 shows that on the MultiNLI dataset, KGAnet's matched accuracy is always higher than that of KIM. Moreover, overfitting occurs later than for KIM.

Therefore, with respect to both training performance and test accuracy, our method feeds more knowledge to the NLI model.

4.4.2 Applicability

To verify the applicability of KGAnet, we added it to multiple NLI models in this experiment. We used DecAtt [13], BiMPM [17], and ESIM [16] as NLI model examples,

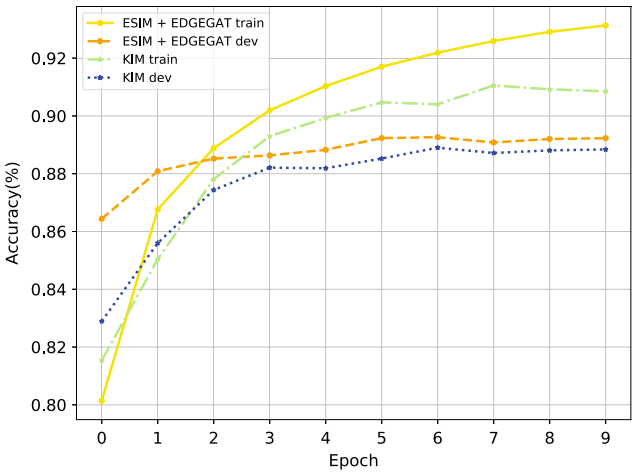


Fig. 4 Training performance of our model (ESIM + KGAnet) and KIM on the SNLI dataset

as shown in Tables 2 and 3. They are all previously proposed NLI models that perform very well but do not incorporate external knowledge. When we incorporate KGAnet, they all improve to some extent. On the SNLI dataset, DecAtt's test accuracy increased by 0.8–86.53%, BiMPM's test accuracy increased by 0.5%, and ESIM's test accuracy increased by 1.0%. Moreover, on the MultiNLI dataset, DecAtt's mismatched set accuracy increased by 0.8–74.96%, BiMPM's mismatched set accuracy increased by 0.6%, and ESIM's mismatched set accuracy increased by 0.7%. These results verify that other NLI

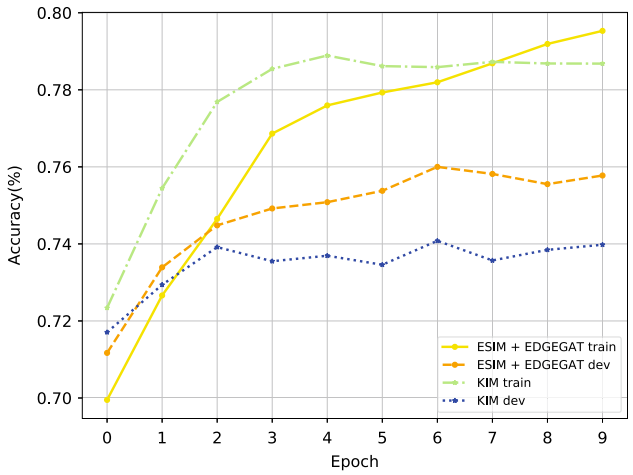


Fig. 5 Training performance of our model (ESIM + KGAnet) and KIM on the MultiNLI dataset

Table 4 Test accuracy comparison of GAT and KGAnet

Model	Train	Dev	Test
ESIM + GAT	91.90 ± 0.20	88.91 ± 0.05	88.24 ± 0.01
ESIM + KGAnet	92.48 ± 0.17	89.26 ± 0.23	88.92 ± 0.04

Table 5 Test accuracy comparison of GAT and KGAnet on the MultiNLI dataset

Model	Train	Matched	Mismatched
ESIM + GAT	81.30 ± 0.05	75.89 ± 0.08	73.95 ± 0.03
ESIM + KGAnet	82.15 ± 0.15	76.03 ± 0.07	74.22 ± 0.02

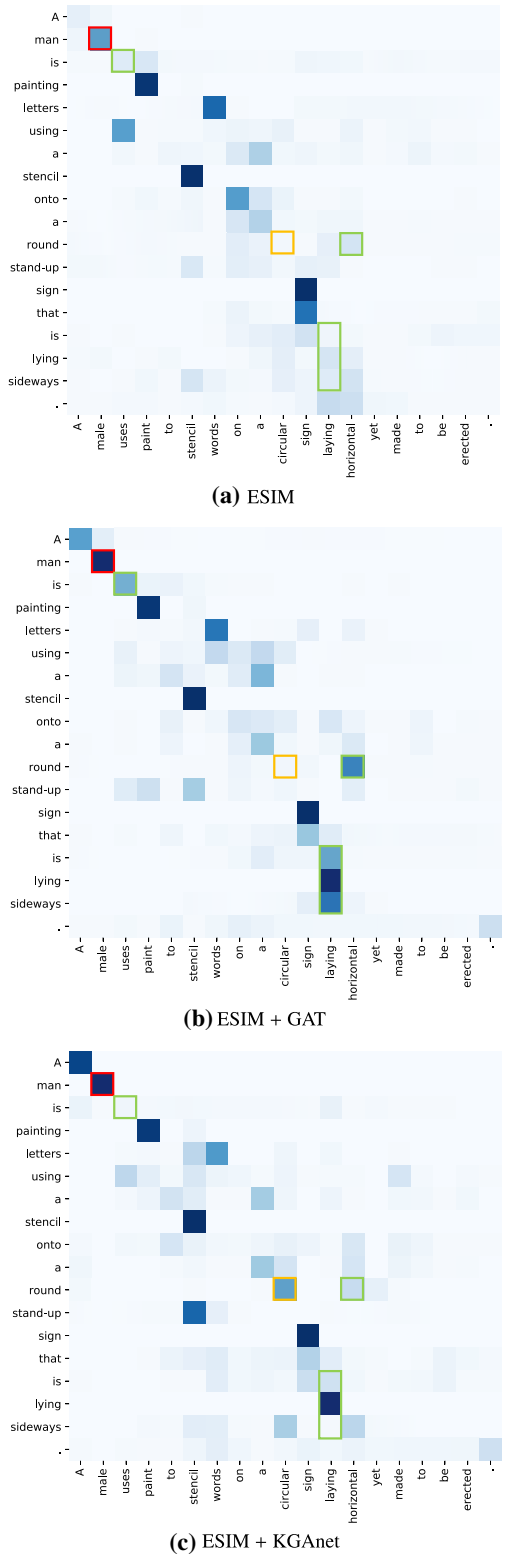
models can access external knowledge through KGAnet without changing their internal structure.

4.5 Comparison of GAT and KGAnet

In Sect. 3.1, we considered the importance of entity relationships based on GAT when we designed KGAnet.

Therefore, we individually added GAT and KGAnet to the same base model (ESIM) to verify whether the relationship is important. The comparison results are shown in Tables 4 and 5. The results show that KGAnet yields a substantially larger improvement than GAT.

Moreover, Fig. 6 shows the attention weights of the three models for an example from the SNLI test set. ESIM + KGAnet successfully predicts it, whereas ESIM + GAT and ESIM fail in this instance. The attention weights map shows the degree of association between the words from the premise and hypothesis in the NLI model. The word-pairs corresponding to a higher degree of association are indicated by darker cells. Whether it is



(a) ESIM

(b) ESIM + GAT

(c) ESIM + KGAnet

Fig. 6 Visualization of cross-attention weights. The left side of the attention map is the premise, and the bottom is the hypothesis. Darker cell colors indicate a higher degree of association between two words. The cells outlined in red indicate the common contribution of KGAnet and GAT. The cells outlined in green indicate noise caused by GAT, and the cells outlined in orange refer to the contribution made by KGAnet itself (color figure online)

necessary to consider the importance of the relationship can be determined by observing the difference between the attention maps. We hence have the following findings:

First In Fig. 6b, c, *man* is matched with *male*, but in Fig. 6a, they are not related. We found in WordNet that *man* and *male* have the common neighbor *lover*. This indicates that GAT and KGAnet capture not only direct relationships, but also the relationships between two indirectly connected nodes.

Second There are no direct or indirect relationships between $\langle is, uses\rangle$, $\langle is, laying\rangle$, $\langle lying, laying\rangle$, $\langle sideways, laying\rangle$, and $\langle round, horizontal\rangle$ in WordNet, which indicates that in Fig. 6b, GAT produces a lot of noise. In contrast, in Fig. 6c, we can see that only the word-pair $\langle lying, laying\rangle$ exists. We believe it is probable that the relationship importance function in KGAnet filters information about some unimportant neighbor entities in the subgraph.

Third The orange outline indicates the contribution $\langle round, circular\rangle$, which has an association only in Fig. 6c. This demonstrates that in KGAnet, the relationship synonymy between $\langle round, circular\rangle$ is indeed added to the calculation and has a certain influence on the word-pair relationships.

In summary, KGAnet reduces the noise, strengthens the association of words-pairs, and greatly improves the effect of NLI model because it considers the importance of the relationship.

5 Conclusion and future work

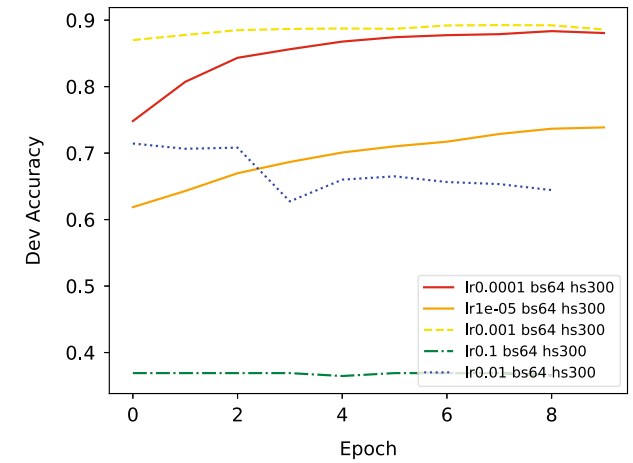
In this paper, we proposed a novel framework to incorporate external knowledge into an NLI model, i.e., by jointly training the proposed KGAnet model and NLI models. With respect to the previous work (KIM), we use a layer of graph neural network to implement the function of introducing external knowledge and there is no preprocessing work on the graph data. And in the experiment, it is verified that KGAnet more sufficiently exploits external knowledge and is more flexible in applying to multiple NLI models.

Although KGAnet has achieved improvements to some extent, some problems remain to be solved. For instance, KGAnet greatly increases the input of the NLI model, which means we must reduce the width of the NLI model during experiments. Of course, if hardware resources are not limited, increasing the scale of the entire graph may yield better results. In addition, only a single-layer KGAnet was considered in this study. In future work, we plan to explore the influence of a k-layer KGAnet and explore more possibilities for introducing external knowledge.

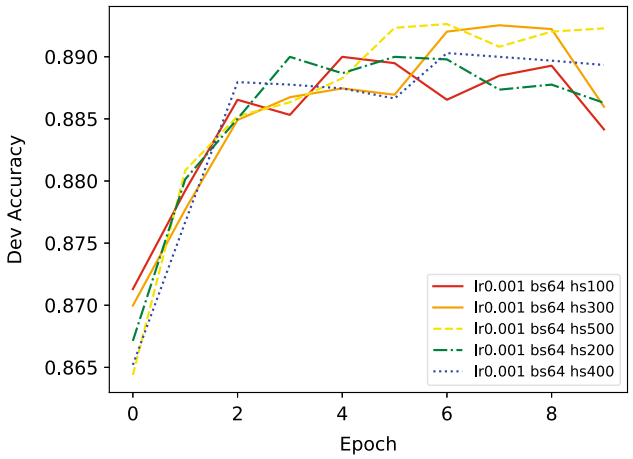
Acknowledgements This work is supported by the National Natural Science Foundation of China (No. 61902034) and Engineering Research Center of Information Networks, Ministry of Education.

Appendix

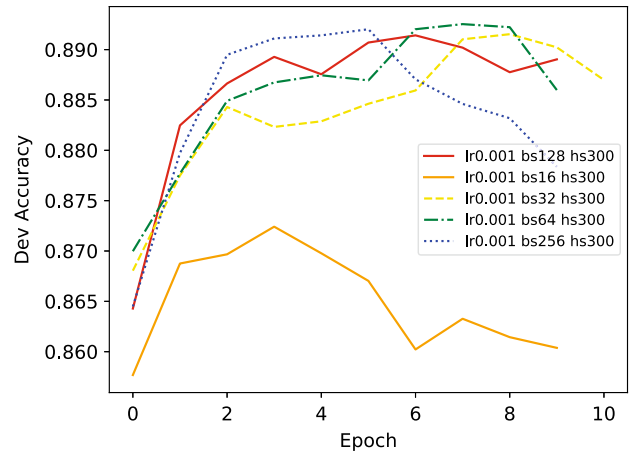
See Fig. 7.



(a) Learning Rate



(b) Hidden Size



(c) Batch Size

Fig. 7 Effects of three hyperparameters (batch size, hidden size, learning rate) on SNLI dataset

References

1. Bowman SR, Angeli G, Potts C, Manning CD (2015) A large annotated corpus for learning natural language inference. arXiv preprint arXiv:1508.05326
2. Williams A, Nangia N, Bowman S (2018) A broad-coverage challenge corpus for sentence understanding through inference. In: Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, vol 1 (long papers). Association for Computational Linguistics, pp 1112–1122
3. Bromley J, Guyon I, Lecun Y, Säckinger E, Shah R (1993) Signature verification using a "siamese" time delay neural network. Int J Pattern Recognit Artif Intell 7(04):669–688
4. Talman A, Yli-Jyrä A, Tiedemann J (2018) Natural language inference with hierarchical bilstm max pooling architecture. arXiv preprint arXiv:1808.08762
5. Nie Y, Bansal M (2017) Shortcut-stacked sentence encoders for multi-domain inference. In: Proceedings of the 2nd workshop on evaluating vector space representations for NLP
6. Shen T, Zhou T, Long G, Jiang J, Wang S, Zhang C (2018) Reinforced self-attention network: a hybrid of hard and soft attention for sequence modeling. arXiv preprint arXiv:1801.10296
7. Im J, Cho S (2017) Distance-based self-attention network for natural language inference. arXiv preprint arXiv:1712.02047
8. Yoon D, Lee D, Lee S (2018) Dynamic self-attention: computing attention over words dynamically for sentence embedding. arXiv preprint arXiv:1808.07383
9. Sabour S, Frosst N, Hinton GE (2017) Dynamic routing between capsules. In: Advances in neural information processing systems, pp 3856–3866
10. Mou L, Men R, Li G, Xu Y, Zhang L, Yan R, Jin Z (2015) Natural language inference by tree-based convolution and heuristic matching. arXiv preprint arXiv:1512.08422
11. Vendrov I, Kiros R, Fidler S, Urtasun R (2015) Order-embeddings of images and language. arXiv:1511.06361
12. Shen T, Zhou T, Long G, Jing J, Pan S, Zhang C (2017) DiSAN: directional self-attention network for RNN/CNN-free language understanding. arXiv:1709.04696
13. Parikh A, Täckström O, Das D, Uszkoreit J (2016) A decomposable attention model for natural language inference. In: Proceedings of the 2016 conference on empirical methods in natural language processing, pp 2249–2255
14. Gong Y, Luo H, Zhang J (2017) Natural language inference over interaction space. arXiv preprint arXiv:1709.04348
15. Tan C, Wei F, Wang W, Lv W, Zhou M (2018) Multiway attention networks for modeling sentence pairs. In: IJCAI, pp 4411–4417
16. Chen Q, Zhu X, Ling Z, Wei S, Jiang H, Inkpen D (2016) Enhanced lstm for natural language inference. arXiv preprint arXiv:1609.06038
17. Wang Z, Hamza W, Florian R (2017) Bilateral multi-perspective matching for natural language sentences. arXiv preprint arXiv:1702.03814
18. Kim S, Hong JH, Kang I, Kwak N (2018) Semantic sentence matching with densely-connected recurrent and co-attentive information. arXiv preprint arXiv:1805.11360
19. Miller GA (1995) WordNet: a lexical database for English. Commun ACM 38(11):39–41
20. Chen Q, Zhu X, Ling ZH, Inkpen D, Wei S (2017) Neural natural language inference models enhanced with external knowledge. In: Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: long papers), vol 1, pp 2406–2417
21. Velickovic P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y (2017) Graph attention networks. arXiv preprint arXiv:1710.10903 1(2)
22. Sen P, Namata G, Bilgic M, Getoor L, Galligher B, Eliassi-Rad T (2008) Collective classification in network data. AI Mag 29(3):93–93

23. Lecun Y, Bengio Y (1995) Convolutional networks for images, speech, and time-series. In: The handbook of brain theory and neural networks. MIT Press
24. Atwood J, Towsley D (2016) Diffusion-convolutional neural networks. In: Proceedings of the 30th international conference on neural information processing systems. Curran Associates Inc., pp 2001–2009
25. Duvenaud D, Maclaurin D, Aguilera-Iparraguirre J, Gómez-Bombarelli R, Hirzel T, Aspuru-Guzik A, Adams RP (2015) Convolutional networks on graphs for learning molecular fingerprints. In: Advances in neural information processing systems, vol 2015, pp 2224–2232
26. Niepert M, Ahmed M, Kutzkov K (2016) Learning convolutional neural networks for graphs. In: Proceedings of the 33rd international conference on international conference on machine learning, vol 48. JMLR.org, pp 2014–2023
27. Hamilton WL, Ying R, Leskovec J (2017) Inductive representation learning on large graphs. arXiv:1706.02216
28. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
29. Bruna J, Zaremba W, Szlam A, LeCun Y (2013) Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203
30. Henaff M, Bruna J, LeCun Y (2015) Deep convolutional networks on graph-structured data. arXiv preprint arXiv:1506.05163
31. Defferrard M, Bresson X, Vandergheynst P (2016) Convolutional neural networks on graphs with fast localized spectral filtering. arXiv:1606.09375
32. Levie R, Monti F, Bresson X, Bronstein MM (2017) Cayleynets: graph convolutional neural networks with complex rational spectral filters. arXiv preprint arXiv:1705.07664
33. Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907
34. Guan J, Wang Y, Huang M (2019) Story ending generation with incremental encoding and commonsense knowledge. In: The thirty-third AAAI conference on artificial intelligence, AAAI 2019. The thirty-first innovative applications of artificial intelligence conference, IAAI 2019. The ninth AAAI symposium on educational advances in artificial intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27–February 1, 2019, pp 6473–6480. https://aaai.org/ojs/index.php/AAAI/article/view/4612
35. Speer R, Chin J, Havasi C (2017) Conceptnet 5.5: an open multilingual graph of general knowledge. In: Proceedings of the thirty-first AAAI conference on artificial intelligence, February 4–9, 2017, San Francisco, California, USA, pp 4444–4451. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14972
36. Zhou H, Young T, Huang M, Zhao H, Xu J, Zhu X (2018) Commonsense knowledge aware conversation generation with graph attention. In: Proceedings of the twenty-seventh international joint conference on artificial intelligence, IJCAI 2018, July 13–19, Stockholm, Sweden, pp 4623–4629. https://doi.org/10.24963/ijcai.2018/643
37. Lin BY, Chen X, Chen J, Ren X (2019) Kagnet: knowledge-aware graph networks for commonsense reasoning. CoRR abs/1909.02151. http://arxiv.org/abs/1909.02151
38. Wang X, He X, Cao Y, Liu M, Chua T (2019) KGAT: knowledge graph attention network for recommendation. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery and data mining, KDD 2019, Anchorage, AK, USA, August 4–8, 2019, pp 950–958. https://doi.org/10.1145/3292500.3330989
39. Wang X, Kapanipathi P, Musa R, Yu M, Talamadupula K, Abdelaziz I, Chang M, Fokoue A, Makni B, Mattei N et al (2019) Improving natural language inference using external knowledge in the science questions domain. Proc AAAI Conf Artif Intell 33:7208–7215
40. Annervaz K, Chowdhury SBR, Dukkipati A (2018) Learning beyond datasets: knowledge graph augmented neural networks for natural language processing. In: Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, vol 1 (long papers), pp 313–322
41. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980
42. Pennington J, Socher R, Manning C (2014) Glove: global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543
43. Bordes A, Usunier N, Garcia-Durán A, Weston J, Yakhnenko O (2013) Translating embeddings for modeling multi-relational data. In: Proceedings of the 26th international conference on neural information processing systems, vol 2. Curran Associates Inc., pp 2787–2795