Tentative Graph Neural Network models for Multi-hop Reasoning

Nanyang Tang (z5103095)
School of Computer Science
University of New South Wales

Abstract

Multi-hop QA across multiple documents has been a new challenge since it requires several supporting sentences to deduce the target answer. In this report, we propose a sentence graph(SG) model to solve this problem, meanwhile, given the good performance of the Cognitive Graph in Hotpot dataset, the Cognitive Graph model is further explored by adding a bi-attention and a transformer layer. The SG model is based on the graph where sentences are regarded as nodes and entities relationship such as Person, Location and Date are denoted as edges. In addition, we use Bert pre-trained model to help extract semantic information, and co-attention graph algorithm(CO-GNN) and graph attention algorithm(GAT) to accumulate evidences on the sentence graph. In this way, it could be showed that SG-model seems to be not sensitive to Bert pre-trained model, however, CO-GNN and GAT could help SG-model to find answers and supporting sentences, respectively. Additionally, for Cognitive Graph, after adding a bi-attention layer and a transformer layer separately, the Cognitive Graph model has surpassed its original achievement on the F1 and EM metrics.

Introduction

The machine reading comprehension has been a prevalent topic in natural language processing area. Deep learning models like Bert (Devlin et al. 2018) , XLNet (Yang et al. 2019), Bidaf (Seo et al. 2016) and et al, particularly, for these pre-trained models such as Bert, XLNet and GPT2 (Radford et al. 2019) has achieved significant advancement even surpassed human in multiple NLP datasets including SQuAD (RRajpurkar et al. 2016) and GLUE (Wang et al. 2018). However, a considerable number of previous works made efforts on single paragraph question answering(QA) (Liu et al. 2017; Wang et al. 2017) and few experiments focus on the multi-hop question answering(QA). This means that many answers are obtained without reasoning. In fact, according to the logic of human to answer questions, the answer is obtained according to a series of supporting evidences which exit in relative sentences or entities. Therefore, some QA datasets such as WikiHop (Welbl et al. 2018), ComplexWeb Questions (Talmor and Berant 2018) as well as HotpotQA (Yang et al. 2018) are proposed to address this issue.

Copyright © 2019, Nanyang Tang

For multi-hop questions, there are two main challenges. The first challenge is to identify the supporting sentences. Since there are much irrelevant information in a document, it is difficult to extract the relationship between these sentences in the context without the prior knowledge. The second challenge is to predict the final answer from these supporting sentences. This is because even if we obtain the supporting sentences, how to deduce the final result is still a problem. In order to tackle with above issues, the graph neural network is increasingly applied to accumulate the corresponding information (Nishida et al. 2019; Feldman and El-Yaniv 2019; Ding et al. 2019; Xiao et al. 2019), nevertheless, these models all employ entities as nodes to construct their relevant graph, which may cause implicit reasoning. This is because these entities may not express explicit information semantically and syntactically.

In this report, we study two models for solving above two multi-hop problems. One of them is called sentence graph(SG) model which takes sentences as nodes and the entity relationship as edges to frame a sentence-based graph. In this model, we assume the answer derives from a series of supporting sentences in a document. Consequently, we establish a relationship graph relying on sentences as nodes and the entity relationship such as Location, Person, Organisation, NORP(Nationalities or religious or political groups.), GPE(Countries, cities, states.) and TITLE(the title of each paragraph) as edges. The reason why we choose this structure is that for human, sentences normally contain more semantic information than entities. Then, like the Cognitive Graph neural network(CGN) (Ding et al. 2019) and Dynamically Fused Graph Network(DFGN) (Xiao et al. 2019), the pre-trained Bert model is also utilised to generate embeddings with relevant information required by GNN. Next, we put the embedding of each sentence to the Graph Neural Network(GNN) to enable the model to extract the relationship of graph. This means the prior knowledge of the document would be introduced to the model. In this way, a information-fusing method (COG) based on co-attention (Zhong et al. 2019) is provided, and the GAT (Veličković et al. 2017) is applied into the SG-model. These GNNs would help reduce the noisy information and obtain higher accurate answers.

The second model is a further ablation study for Cognitive graph(CGN) (Ding et al. 2019). The Cognitive graph

model uses titles and the answer in a document as the node to construct a directed graph. It uses Bert to extract question-relevant entities and answer candidates from paragraphs and encodes their semantic information, and then a GNN (Battaglia et al. 2018) is utilised to do reasoning procedure over the graph and find clues to help Bert to better predict supporting sentences. In this report, we attempt to add a bi-attention layer which is designed to emphasize the relation between the paragraph and the question to this model. In addition, although the Bert could generate semantic information, when we accumulate these embeddings including semantic representation from the output of Bert, they do not establish the potential entity relationship. As a result, a transformer (Vaswani et al. 2017) layer is implemented in the CGN model to make each entity have the potential relationship.

Our research contributions are as follows:

- we propose the novel SG-model for the multi-hop text-based QA problem.
- we show that the graph based on sentences could help reach higher accurate results.
- we estimate the performance of the Bi-attention layer and the Transformer layer on Cognitive graph

Related Work

HotpotQA dataset:

QA tasks could be Knowledge-based(KBQA), text-based(TBQA), mixed and others. This means that the core of research on machine reading comprehension has transferred to QA tasks with reasoning from simple cloze-style tasks (Hermann et al. 2015; Hill et al. 2015). In this report, the HotpotQA (Yang et al. 2018) which is TBQA would be studied. This is because it can estimate the capability of answer extraction and reasoning them. Different from the single hop question like SQuAD (RRajpurkar et al. 2016), multi-hop questions require multiple facts comparative to the question to reason the answer. Hence, many techniques based on text and IR are hard to be applied into multi-hop issues. In addition, compared with the SQuAD having free format of answer, HotpotQA limits their answer to be the format like entity and boolean format.

GNN for NLP:

Due to the reasoning property of multi-hop, GNN has widely been applied to multi-hop QA problems and significant success has been achieved. Various GNNs are proposed to solve QA tasks requiring reasoning, for example, the graph convolution network (Kipf and Welling 2016), graph attention network(GAT) (Veličković et al. 2017) and graph recurrent network (Song et al. 2018) . Owing to these base networks, many new models are proposed to tackle with multi-hop issues. For example, Heterogeneous Graphs (Tu et al. 2019) and Entity Graph Convolutional Networks (De Cao, Aziz and Titov. 2018) has achieved good scores in the WikiData set (Welbl et al. 2018), where both of them used the graph convolution network. In addition, the DFGN (Xiao et al. 2019) derived from GAT also has the excellent performance in the Hotpot dataset. However, these GNNs rely on the graph regarding entities as nodes, and edges of the graph are connected by variously implicit and explicit relations such as synonyms and co-reference.

Cognitive graph:

Cognitive graph (Ding et al. 2019) has achieved great success in HotpotQA problem. It depends on each title and the answer in a document as nodes to construct a directed graph. This means a title only connects with an answer with a directed edge from title to answer. In addition, it applied Bert to extract semantic features and use the GNN (Battaglia et al. 2018) to add the prior information to the model. It could be noticed that this model would dynamically find supporting entities and answer so this would considerably reduce the risk of missing answer. According to the cognitive graph, we attempt to employ a bi-attention layer to improve its performance cause for long contexts, bi-attention is a good way to boost the information interaction. Similarly, given the fact that the relation between entities cannot be represented in the Bert model, so a transformer (Vaswani et al. 2017) layer is utilised to implicitly represent the relations.

SG-model

Data preprocessing

In this HotpotQA data training set, there are 90447 instances and each instance includes at most 10 paragraphs. It could be found that 92% of supporting sentences exist in the first three sentences in each paragraph. In addition, around 90% of sentences contain at most 50 words. As a result, for each question, we select all paragraphs, and for each paragraph, we select the first three sentences with the first 50 words, so the total length for a paragraph is 150 words. If the length of a paragraph cannot achieve 150, it would be padded to complement for the consistency.

In the SG-model, the Spacy (Strubell et al. 2017) is utilised to extract entities in each sentence, where such entities as PERSON(People, including fictional.), NORP(Companies, agencies, institutions, etc.), ORG(organization), GPE(Countries, cities, states.), LOC(Non-GPE locations, mountain ranges, bodies of water.), DATE(Absolute or relative dates or periods.) and the title of each paragraph would be considered as the interacting entities to construct the sentence graph. Fig1 shows the process extracting entities. After completing the construction of graph, sentences would be semantically and syntactically connected, and the rule of connecting derives from synonym, co-reference, and relation of identity. Finally, the propagation of GNN would be implemented in this graph three times.

The input of model

For each instance, a sub-network is selected to find relevant sentences. The sub-network is based on the pre-trained Bert model (Devlin et al. 2018) and the input takes a question and a paragraph. Nevertheless, different from the input of Bert for SQuAD, in order to keep the consistency of length

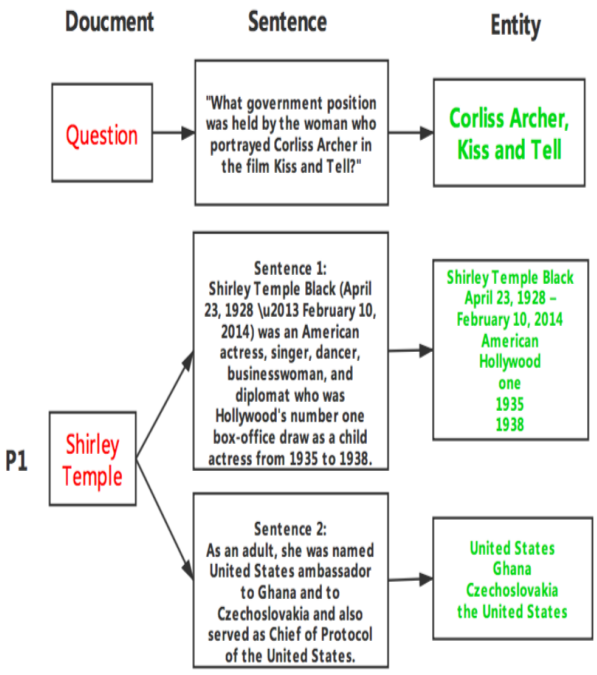

Figure 1: The entities in paragraphs. The red words indicate titles of paragraphs, the black sentences indicate corresponding sentences in a paragraph and the green words indicate the entities from each sentence. P1 indicates the first paragraph in a document.

of each sentence, the pad should be added to the end of each sentence. The example of input is as follows:

$$\underbrace{[CLS]question}_{sentence1+PAD} + \underbrace{[SEP] + paragraph + [SEP]}_{sentence2+PAD}$$

The outputs of Bert are denoted as $O \in \mathbb{R}^{N\times S\times W}$, where N is the number of paragraphs in an instance, S is the number of sentence in a paragraph and W is the number of words in a sentence.

The loss function to fine-tune

Due to the fact that Bert model is strong for most NLP tasks, hence, we could directly use the output of Bert to fine tune Bert model, after that, it could be used in downstream tasks. The properties of answer and supporting sentences are different. In this report, we regard the answer and supporting sentences as the labels to make loss function with the output of Bert. This is because answers normally determined by question is an entity or the boolean answer, nevertheless, the supporting sentences rely on the paragraphs. Therefore, we separately predict answers and support sentences. In this way, the output of Bert is $O \in \mathbb{R}^{N\times S\times W\times H}$, where N is the number of paragraphs in an instance, S is the number of sentence in a paragraph, W is the number of words in a sentence and H is the hidden size. Then, the computing process is as follows:

$$S_{start} = Linear(O) \in \mathbb{R}^{N\times S\times W} \tag{1}$$

$$S_{end} = Linear(O) \in \mathbb{R}^{N\times S\times W} \tag{2}$$

$$A_{start} = Linear(O) \in \mathbb{R}^{N\times S\times W} \tag{3}$$

$$A_{end} = Linear(O) \in \mathbb{R}^{N\times S\times W} \tag{4}$$

After we obtain the S_{start}, S_{end}, A_{start} and A_{end}, we could use cross-entropy to compute probabilities of starting position of supporting sentences, end position of supporting sentences, starting position of answer and end position of answer.

$$P_j^i[k] = \log \frac{e^{[j_i][k]}}{\sum_k e^{j_i}} \tag{5}$$

where **j** is one of S(supporting sentences) and A(answer), **i** is one of start and end and **k** is the length of each input in this model. The the loss could be made.

$$loss = \frac{1}{2}\sum_j - \sum_i Y_j^i \times P_j^i \tag{6}$$

where Y_j^i is the target labels of j_i.

Graph Neural network

In this part, we use these entities extracted in Fig1 to connect sentences following such relations as synonym, co-reference and identity in a document. Fig2 demonstrates the graph. Next, two GNN models are employed to test their performances on results. The first GNN,CO-GNN, is inspired from co-attetion (Ding et al. 2019), which could fuse the sentence information in word level well. The second GNN is GAT (Veličković et al. 2017), which can be utilised to absorb the graph information in sentence level.

CO-GNN After framing the relationship graph among sentences and completing the fine-tuning, we obtain the $O \in \mathbb{R}^{N\times S\times W\times H}$ from Bert. Then, for each sentence, we attempt to extract information from its neigbours. In addition, due to the fact that each paragraph has one question, we use a linear layer to obtain an unified representation for the question in each document. The process is demonstrated in Algorithm 1.

In Algorithm 1, we firstly extract question embeddings $Q_o \in \mathbb{R}^{1\times W\times H}$ from $O \in \mathbb{R}^{N\times S\times W\times H}$.

$$Q_i = O[:, q_{len}, :, :] \in \mathbb{R}^{N\times W\times H} \tag{7}$$

$$Q_o = w_q \times Q_i \in \mathbb{R}^{W\times H} \tag{8}$$

where $w_q \in \mathbb{R}^{1\times N}$, $q_{len} = 0$ and $O \in \mathbb{R}^{N\times S\times W\times H}$; Then, for each sentence, we frame a matrix Am containing itself and their corresponding sentences to do a co-attention (Zhong et al. 2019) . In this way, the sentence selected is placed in the first row in Am as the central sentence, and other sentences are affiliated sentences. In order to achieve this aim, we construct a dictionary for each sentence where the key and values are their indexes in their document. For example, "I am a hero" is the first sentence

in the third paragraph, so its index is "$(2, 0)$". Additionally, the question in a document is denoted as $(-1, 0)$. After $Am \in \mathbb{R}^{Node_{\#} \times W \times H}$ is filled, we would put it into co-attention. First of all, the central sentence and these affiliated sentences related to it would be selected as follows:

$$A_r = Am[0] \in \mathbb{R}^{W \times H}, \quad A_i = Am[i] \in \mathbb{R}^{W \times H} \tag{9}$$

where A_r represents the central sentence and A_i is the affiliated sentence. Then, A_r and A_i could be calculated as:

$$A^i_{qs} = A_r \times A_i^T \in \mathbb{R}^{W \times W} \tag{10}$$

where T represents matrix transpose. In A^i_{qs}, each entry indicates how related two sentences in word level are. Next, the attention would occur between the central sentence and the affiliated sentence.

$$C^i_q = softmax({A^i_{qs}}^T) \times A_r \in \mathbb{R}^{W \times H} \tag{11}$$

$$C^i_s = softmax(A^i_{qs}) \times A_i \in \mathbb{R}^{W \times H} \tag{12}$$

Then, we use a Linear layer to encode A^i_{qs} to obtain a result, and concatenate C^i_s and the result to obtain a comprehensive consequence S^i_{ca}.

$$D^i_s = Linear(softmax(A^i_{qs}) \times C^i_q) \in \mathbb{R}^{W \times H} \tag{13}$$

$$S^i_{ca} = [C^i_s; D^i_s] \in \mathbb{R}^{W \times 2H} \tag{14}$$

In this way, we expect S^i_{ca} to represent the importance of i_{th} affiliated sentence to the central sentence. Hence, a perceptron is introduced to generate a vector representation for the i_{th} affiliated sentence.

$$a_i = softmax(MLP(S^i_{ca})) \in \mathbb{R}^{W \times 1} \tag{15}$$

where MLP is a neural network with two layers. After each sentence has their a_i, we apply the attention mechanism to know how related each affiliated sentence and the central sentence are.

$$\beta_{0,i} = LeakyRelu(W_b^T[a_0; a_i]) \in \mathbb{R} \tag{16}$$

$$\alpha_i = \frac{exp(\beta_{0,i})}{\sum_{i=0}^{i=Node_{\#}} exp(\beta_{0,i})} \tag{17}$$

where $W_b \in \mathbb{R}^{2W}$. Finally, we could obtain the consequence containing graph information for the central sentence due to α_i and S^i_{ca}.

$$co_{result} = \sum_{i=0}^{i=Node_{\#}} \alpha_i \times S^i_{ca} \in \mathbb{R}^{1 \times W \times 2H} \tag{18}$$

where $Node_{\#}$ is the number of affiliated sentences with respect to the central sentence.

Graph attention In our experiment, in addition to the CO-GNN, we also attempt to use GAT (Veličković et al. 2017) to SG-model. This method is firstly to decrease the dimension of $O \in \mathbb{R}^{N \times S \times W \times H}$ to represent the document in sentence level.

$$S_l = Linear(O) \in \mathbb{R}^{N \times S \times W} \tag{19}$$

Then, for each sentence, their affiliated sentences would pass the GAT.

$$a_i = S_l[i] \in \mathbb{R}^W, \quad a_j = S_l[j] \in \mathbb{R}^W \tag{20}$$

where a_i represents the vector of central sentence i and a_j represents the vector of affiliated sentence j.

$$\beta_{i,j} = LeakyRelu(W_b^T[a_i; a_j]) \in \mathbb{R} \tag{21}$$

$$\alpha_j = \frac{exp(\beta_{i,j})}{\sum_{j=0}^{j=Node_{\#}} exp(\beta_{i,j})} \tag{22}$$

where α_j represents the proportion of information that will be assigned to the neighbors of central sentence i.

$$co_{result} = \sum_{j=0}^{j=Node_{\#}} \alpha_j \times a_j \in \mathbb{R}^W \tag{23}$$

After all sentences propagate in one of above two GNNs, for SG-model, the consequence of CO-GNN or GAT, O_G, would produce four predictions including S_{start}, S_{end}, A_{start} and A_{start}.

$$S_{start} = Linear(O_G) \in \mathbb{R}^{N \times S \times W} \tag{24}$$

$$S_{end} = Linear(O_G) \in \mathbb{R}^{N \times S \times W} \tag{25}$$

$$A_{start} = Linear(O_G) \in \mathbb{R}^{N \times S \times W} \tag{26}$$

$$A_{end} = Linear(O_G) \in \mathbb{R}^{N \times S \times W} \tag{27}$$

Prediction For predicting the supporting sentences, not like the unique answer, normally a document exists several supporting sentences, therefore, we concatenate the S_{start} and S_{end} to identify which sentences are supporting sentences, where a GRU is employed as an encoder.

$$S_{all} = GRU([S_{start}; S_{end}]) \in \mathbb{R}^{N \times S \times 2W} \tag{28}$$

$$S_t = sigmoid(Linear(S_{all})) \in \mathbb{R}^{N \times S} \tag{29}$$

In our experiment, the $S_t[P][i] \geq 0.5$ is that the i_{th} sentence in P_{th} paragraph is the supporting sentence, otherwise, it is not the supporting sentence. For the prediction of answer, if the type of answer is entity, we could use P^i_j[k] (5) to predict the answer. If the type of answer is "yes-no",

$$S_p = [Linear(S_{all}[P][i]) \quad if \quad S_t[P][i] \geq 0.5] \tag{30}$$

$$P_{yes/no} = sigmoid(Linear(S_p)) \tag{31}$$

When $P_{yes/no}$ is more than 0.3, the answer is yes, otherwise it is no.

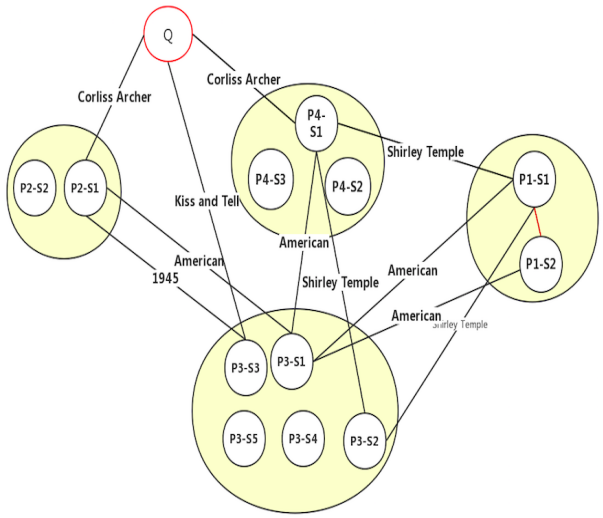

Figure 2: The graph based on sentences as nodes . These nodes in a yellow circle are connected each other. This means they are in the same paragraph. The P delegates paragraph, S is sentence and Q is the question.

Algorithm 1: CO-GNN heuristic

Data: output from Bert $O \in \mathbb{R}^{N\times S\times W\times H}$, graph G
Result: the matrix containing graph information O_G

Initialise a zero matrix $O_G \in \mathbb{R}^{N\times S-1\times W\times H}$;
Q = linear($O[:, q\_len, :, :]$) $\in \mathbb{R}^{1\times W\times H}$; where q_len = 0, cause the question always exists in the head of paragraphs.
for $S_i \in G$ **do**
  $Node_{\#} = len\,(G[S_i])$;
  Initialise a zero matrix Am $\in \mathbb{R}^{Node_{\#}\times W\times H}$;
  if $S_i[0]$ = -1 **then**
    Am[0,:,:] += Q
  else
    Am[0,:,:] = O[S_i[0],S_i[1],:,:]

  for $c_{\#}, c_{node} \in G[S_i]$ **do**
    if $c_{\#} = -1$ **then**
      $Am[c_{\#}, :, :] += Q$
    else
      $Am[c_{\#}, :, :] = O[c_{node}[0], c_{node}[1], :, :]$

  $co_{result} = co\_attention\,(Am) \in \mathbb{R}^{1\times W\times H}$
  if $S_i[0] \neq -1$ **then**
    $O_G[S_i[0], S_i[1], :, :] += co_{result}$
  else
    Q = co_{result}

return $O_G \in \mathbb{R}^{N\times S-1\times W\times H}$;

Cognitive Graph Neural Network(CGN)

In addition to SG-model, we also delve the Cognitive Graph Neural Network (Ding et al. 2019). This neural network is also based on the Bert pre-trained model and the construction of Cognitive graph stems from the fact that each title in the document is considered as an entity. If the answer exists in a paragraph, there is an edge between the corresponding entity and the answer. The process of inference in Cognitive Graph Neural Network is that, according to the Cognitive Graph, $the\ \ question + the\ \ relative\ \ sentence + the\ \ current\ \ paragraph$ is used as the input of Bert model. After it passed from Bert model, the embedding of [CLS] of each paragraph would be selected as the entity embedding. This is because [CLS] is used to do the classification task in Bert. This means the [CLS] has included the summary property for a paragraph. Then, these entity embeddings would passed the GNN algorithm (Battaglia et al. 2018), and the inference would be proceeded after the data is yielded from the propagation of GNN. Although the Cognitive Graph Neural Network has achieved good performance in HOTPOT dataset, there are still some rooms for improvement in this model.

Bi-attention in CGN

Despite the fact that transformers have grasped good semantic and syntactic features, it seems the information of question still would be weakened in transformer encoders. In fact, the most importance factor in a QA problem is the question. Consequently, after data passed Bert model, we try to add a Bi-attention layer to increase the weight of questions. The following calculation shows the process of Bi-attention.

$$query \in \mathbb{R}^{N\times W_q\times H} \quad context \in \mathbb{R}^{N\times W_c\times H} \tag{32}$$

where W_q is the length of $query$ and W_c is the length of $query + context$.

$$query_l = Linear(query) \in \mathbb{R}^{N\times 1\times W_q} \tag{33}$$

$$context_l = Linear(context) \in \mathbb{R}^{N\times W_c\times 1} \tag{34}$$

where two Linear layers are denoted as two encoders for the query and the context, respectively.

$$cross = context \times query^T \in \mathbb{R}^{N\times W_c\times W_q} \tag{35}$$

$$att = query_l + context_l + cross \in \mathbb{R}^{N\times W_c\times W_q} \tag{36}$$

where for $query_l$ and $context_l$, they are a value in $query_l$ add a vector in $context_l$, and for $cross$, it point-wise adds with others. After that, each entry in att delegates how related each word of query and context is.

$$context_{l2} = Linear(context) \in \mathbb{R}^{N\times W_c} \tag{37}$$

$$query_{l2} = Linear(query) \in \mathbb{R}^{N\times W_q} \tag{38}$$

where two Linear layers are denoted as two encoders for the query and the context, respectively.

$$O_1 = softmax(att) \times query_{l2} \in \mathbb{R}^{N \times W_c \times H} \quad (39)$$

$$O_2 = softmax(att.max()) \times context_{l2} \in \mathbb{R}^{N \times W_c \times H} \quad (40)$$

where $max(.)$ denotes column-wise max.

Finally, the output of Bi-attention is $U = [context; O_1; context * O_1; O_1 * O_2] \in \mathbb{R}^{N \times W_c \times 4H}$. Then the U would be applied in the downstream task.

Transformer for CGN

These data consist of N embeddings of [CLS] would pass the GNN (Battaglia et al. 2018) to predict answer. For example, assume $O \in \mathbb{R}^{N \times W \times H}$ is the output of Bert, then

$$semantics = O[:, 0, :] \in \mathbb{R}^{N \times H} \quad (41)$$

, next the $semantics$ would be transferred to GNN to collect graph information. However, prior to that, we assume these embeddings in $semantics$ have no any relation each other. As a result, we use a transformer (Vaswani et al. 2017) with 1 layer and 3 heads to make it have implicit information.

Experiment

Experimental details

We use the training set(90,477 questions) and the distractor setting(2000 questions) as the development set of HotpotQA. In this way, 84% question require multi-hop reasoning and all questions in development set are hard multi-hop cases. In addition, the uncased version of Bert model would be used to fine-tune. The hidden size H is 768, the activation function for CO-GNN is relu, and the activation function for CGN is gelu (Hendrycks and Gimpel 2016). The optimizer is Adam (Kingma, D.P. and Ba, J. 2014) with learning rate of 0.0001, and the probability of dropout is 0.3.

During the stage of graph construction, the Spacy (Strubell et al. 2017) is used to extract named entities in each paragraph, where such entities as PERSON(People, including fictional.), NORP(Companies, agencies, institutions, etc.), ORG(organization), GPE(Countries, cities, states.), LOC(Non-GPE locations, mountain ranges, bodies of water.), DATE(Absolute or relative dates or periods.) and the title of each paragraph would be considered as the connecting entities to construct the sentence graph.

Main Results

SG-model We first show the results of SG-model including only-Bert, CO-GNN and GAT. In addition, the ablation study including a bi-attention layer and a transformer layer is also applied into the CGN.

From F1 and EM metrics in Table 1, we could know, the only Bert model learns nothing. This may be because we limit the length of each sentence, which means these PADs at the end of sentences have adverse effect on Bert pre-trained model so the performance of Bert pre-trained model is comparatively terrible. Another reason why the loss of only Bert model is high is that the training epoch is few. However, when the CO-GNN is added into only Bert model, this model starts to learn some information, particularly, for the supporting sentences. Similarly, after baseline model plus the GAT, the information from answers and supporting sentences is captured, and it achieves good performance in answer metrics. Consequently, it is clear that Graph neural network indeed could have good effect on the multi-hop QA problems. Additionally, the CO-GNN is beneficial for predicting the supporting sentences, and the GAT has good achievement in predicting answer.

CGN It is obvious that compared with SG-model, the Cognitive Graph Neural Network has extremely low loss with 1 epoch. This means it learns well in training sets. From F1 and EM metrics, we could know it also has better performance than the SG-model. In addition, for the ablation study of CGN, Bi-attention is beneficial to the prediction of answer, and the transformer layer also could help learn more information. However, when we combine Bi-attention and Transformer, in spite of the fact that the answer loss and support loss reduce to a relatively low point, it does not perform well compared with the only Bi-attention or only Transformer. This means that combination of Bi-attention and Transformer may lead to an overfitting.

Recommendation

SG-model seems not to have good scores in F1 and EM metrics, this is because we change the format of input of Bert model and set few training epochs as a result of the limit of computing resources. In addition, the length of sentence is limited to 50 words, which may lead to some important information is lost. Therefore, increasing the training epoch and extending the length of sentences may boost the learning of model. Furthermore, for SG-model, it appears Bert model can not work on it since the input of Bert is different from the input of SG-model, which lead to the failure of Bert on SG-model. As a result, exploring a new sentence graph based on the effective input of Bert may enhance the performance of SG-model.

For CGN, it has had good performance in F1 and EM. However, its answer is directly derived from the title entities. Hence, For some questions requiring intermediate entities to reason, it is hard to extract correct information for CGN. As a result, how to add these relaying entities into Cognitive graph may be a direction of improvement.

Conclusion

In conclusion, we introduce a Sentence Graph model(SG-model) to solve the multi-hop problem, meanwhile, we also refine the Cognitive Graph model, and add a Bi-attention layer and a Transformer layer to it, respectively. Specifically, for SG-model, we firstly use the graph neural network based on sentences as entities and prove that different graph computing methods including CO-GNN and GAT could improve the performance of model as various bias. For Cognitive Graph model, we also prove that the bi-attention and the transformer layer could assist to extract the answer entity. In

Table 1: SP-model performance comparison by two evaluation metrics.

	Ans		Sup		Loss	
	F1	EM	F1	EM	ans_loss	sup_loss
Only Bert	0.00	0.00	0.00	0.00	3.540	7.570
Bert+CO-GNN	3.80	3.80	0.00	***12.00***	4.460	8.750
Bert+GAT	***10.00***	***10.00***	0.00	8.86	6.600	11.210

Table 2: CGN-model performance comparison by two evaluation metrics.

	Ans		Sup		Loss	
	F1	EM	F1	EM	ans_loss	sup_loss
Cognitive Graph	42.00	55.13	16.00	58.31	0.54	0.07
CGN+Bi-attention	***44.00***	***57.00***	14.50	57.88	0.16	0.02
CGN+Transformer	42.00	***56.00***	15.50	57.52	0.11	0.01
CGN+Bi-attention+Transformer	39.50	54.10	11.5	55.90	0.11	0

the future, the pre-trained model for SG-model would be explored to enhance the ability to extract answer and support sentences.

References

Battaglia, P.W., Hamrick, J.B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R. and Gulcehre, C. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261.*

De Cao, N., Aziz, W. and Titov, I. 2018. Question answering by reasoning across documents with graph convolutional networks. *arXiv preprint arXiv:1808.09920.*

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805.*

Hermann, K.M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M. and Blunsom, P. 2015. Teaching machines to read and comprehend. understanding. *In Advances in neural information processing systems (pp. 1693-1701).*

Hill, F., Bordes, A., Chopra, S. and Weston, J. 2015. The goldilocks principle: Reading children's books with explicit memory representations. *arXiv preprint arXiv:1511.02301.*

Ding, Ming, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive Graph for Multi-Hop Reading Comprehension at Scale. *arXiv preprint arXiv:1905.05460 (2019).*

Hendrycks, D. and Gimpel, K. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units.

Kingma, D.P. and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

Xiaodong Liu, Yelong Shen, Kevin Duh, and Jian- feng Gao. 2017. Stochastic answer networks for machine reading comprehension. *arXiv preprint arXiv:1712.03556.*

Nishida, K., Nishida, K., Nagata, M., Otsuka, A., Saito, I., Asano, H. and Tomita, J. 2019. Answering while Summarizing: Multi-task Learning for Multi-hop QA with Evidence Extraction. *arXiv preprint arXiv:1905.08511.*

Kipf, T.N. and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907.*

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. and Sutskever, I. 2019. Language models are unsupervised multitask learners. *OpenAI Blog, 1(8).*

Rajpurkar, P., Zhang, J., Lopyrev, K. and Liang, P. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250.*

Seo, M., Kembhavi, A., Farhadi, A. and Hajishirzi, H. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603.*

Song, L., Zhang, Y., Wang, Z. and Gildea, D. 2018. A graph-to-sequence model for AMR-to-text generation. *arXiv preprint arXiv:1805.02473.*

Strubell, E., Verga, P., Belanger, D. and McCallum, A. 2017. Fast and accurate entity recognition with iterated dilated convolutions. *arXiv preprint arXiv:1702.02098.*

Talmor, A. and Berant, J. 2018. The web as a knowledge-base for answering complex questions. *In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computa- tional Linguistics: Human Language Technologies, Volume 1 (Long Papers), volume 1, pages 641–651.*

Tu, M., Wang, G., Huang, J., Tang, Y., He, X. and Zhou, B. 2019. Multi-hop Reading Comprehension across Multiple Documents by Reasoning over Heterogeneous Graphs. *arXiv preprint arXiv:1905.07374.*

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I. 2017. Attention is all you need. *In Advances in neural information processing systems (pp. 5998-6008).*

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P. and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903.*

Wang, W., Yang, N., Wei, F., Chang, B. and Zhou, M. 2017. Gated self-matching networks for reading comprehension and question answering. *In Proceedings of the 55th Annual Meet- ing of the Association for Computational Linguistics (Volume 1: Long Papers), volume 1, pages 189–198.*

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O. and Bowman, S.R. 2017. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461.*

Welbl, J., Stenetorp, P. and Riedel, S., 2018. Constructing datasets for multi-hop reading comprehension across documents. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics, 6, pp.287-302.*

Xiao, Y., Qu, Y., Qiu, L., Zhou, H., Li, L., Zhang, W. and Yu, Y. 2019. Dynamically Fused Graph Network for Multi-hop Reasoning. *arXiv preprint arXiv:1905.06933.*

Feldman, Y. and El-Yaniv, R. 2019. Multi-Hop Paragraph Retrieval for Open-Domain Question Answering. *arXiv preprint arXiv:1906.06606.*

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. and Le, Q.V. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *arXiv preprint arXiv:1906.08237.*

Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W.W., Salakhutdinov, R. and Manning, C.D 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600.*

Zhong, V., Xiong, C., Keskar, N.S. and Socher, R. 2019. Coarse-grain fine-grain coattention network for multi-evidence question answering. *arXiv preprint arXiv:1901.00603.*