Relation-aware Bidirectional Path Reasoning for Commonsense Question Answering

Junxing Wang, Xinyi Li, Zhen Tan, Xiang Zhao, Weidong Xiao
National University of Defense Technology, China
{junxingwang,xinyili,tanzhen08a,xiangzhao,wdxiao}@nudt.edu.cn

Abstract

Commonsense Question Answering is an important natural language processing (NLP) task that aims to predict the correct answer to a question through commonsense reasoning. Previous studies utilize pre-trained models on large-scale corpora such as BERT, or perform reasoning on knowledge graphs. However, these methods do not explicitly model the *relations* that connect entities, which are informational and can be used to enhance reasoning. To address this issue, we propose a relation-aware reasoning method. Our method uses a relation-aware graph neural network to capture the rich contextual information from both entities and relations. Compared with methods that use fixed relation embeddings from pre-trained models, our model dynamically updates relations with contextual information from a multi-source subgraph, built from multiple external knowledge sources. The enhanced representations of relations are then fed to a bidirectional reasoning module. A bidirectional attention mechanism is applied between the question sequence and the paths that connect entities, which provides us with transparent interpretability. Experimental results on the CommonsenseQA dataset illustrate that our method results in significant improvements over the baselines while also providing clear reasoning paths.

1 Introduction

Commonsense Question Answering (CQA) is a task that requires machines to not only understand the question, but also infer through external knowledge. For example, to answer the question in Figure 1 ("What do people typically do while playing guitar?"), it is necessary to discover the connection between the evidence "playing guitar" and "singing". Although seemingly trivial for humans, knowledge like this is difficult to learn for natural language understanding (NLU) models.

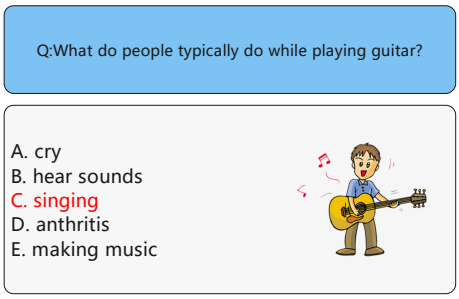

Figure 1: An example of CommonsenseQA.

Pre-trained language models (PTLMs) with distributed representations, such as BERT, RoBERTa, XLNet and ELECTRA(Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019; Clark et al., 2020), have performed well on many Natural Language Processing (NLP) tasks. However, when answering commonsense questions, these models fail to explicitly recover the reasoning process. Specifically, for the question in Figure 1, humans figure out the correct answer by inferring from the relation between "playing guitar" and "singing", while PTLMs only provide a predicted answer and can not explain the inference process. They are thus lacking in transparency and interpretability. Apparently, there is still a wide gap between the PTLMs and human level performance in CQA tasks.

Since merely relying on pre-trained large language models can not utilize commonsense knowledge for reasoning, a straightforward solution is to model the paths(Lin et al., 2019) in the knowledge graph "that connect pairs of concepts". KagNet(Lin et al., 2019) encodes the paths extracted from the knowledge graph, and further provides interpretability through the attention mechanism. (Lv et al., 2020) extract evidence from heterogeneous external knowledge bases and reason based on the graph neural network. These methods have managed to help the model learn background knowledge, especially the entities and structures in the knowledge graph. But they either completely ig-

Proceedings of the 25th Conference on Computational Natural Language Learning (CoNLL), pages 445–453
November 10–11, 2021.

nore relations or learn simple representations of relations through pre-trained models(TransE(Wang et al., 2014)). These methods keep the pre-trained relation representations unchanged when training their question answering model. And hence their models do not fully learn the background knowledge.

Relations are an integral part of knowledge and should not be ignored. Therefore, enabling the model to *learn relations* helps to fully utilize the background knowledge in the CQA task. We propose a relation-aware bidirectional reasoning framework **RaB-PR**[1] for CQA. The framework first extracts evidence from both knowledge graphs and Wikipedia to form a multi-source subgraph. The graph is then integrated into a relation-aware graph convolutional network to jointly learn representations of nodes and *relations*. Furthermore, we introduce a new attention mechanism between the paths and the question to selectively derive salient information for final predictions. The relation-aware graph convolutional networks on multi-source subgraphs enables our model to comprehensively capture the rich evidence contained in external knowledge bases (especially the *relations*), and the attention mechanism brings better interpretability for inference. The experiments on the CommonsenseQA dataset indicate that our method achieves superior performances over the baseline.

Our contributions of this paper can be summarized as follows:

- We introduce a relation-aware graph convolutional network, CompGCN, which can learn nodes and *relations* in multi-relation graphs based on *composition*(definition in 4.2).
- We propose a relation-aware bidirectional reasoning framework, which can explain the reasoning process of commonsense question answering based on paths in the subgraph.
- Our bidirectional reasoning method exhibits a significant performance improvement on the CommonsenseQA dataset over previous models.

The rest of this paper is organized as follows. In section 2, we give a brief review of the related work. In section 3, we first provide a formal definition of commonsense question answering, and then briefly introduce the framework of our method. In section 4, we introduce the relation-aware bidirectional path learning method in detail. Afterwards, section 5 presents the experiments on the CommonsenseQA dataset with analysis. Finally, the conclusion and future work are given in section 6.

[1]We will open source our code upon acceptance of our submission.

2 Related Work

Our work is related to commonsense reasoning and graph neural networks. Below, we introduce the representative work and how our work differs from theirs.

2.1 Commonsense Reasoning

Unlike the typical question answering task(Chen et al., 2017; Wang et al., 2019; Wu et al., 2017), Commonsense Reasoning is a task to infer the correct answer and identify implicit causes and effects combined with external commonsense knowledge. There is a recent surge of novel large-scale datasets(Sap et al., 2019; Rashkin et al., 2018; Ostermann et al., 2019; Zellers et al., 2018, 2019; Mostafazadeh et al., 2016) for testing machine commonsense with various focuses. Among them, a recently proposed dataset CommonsenseQA(Talmor et al., 2019) attracts much research attention. Entity-GCN(Cao et al., 2019a) extracts an entity-graph from the context of the question, and models the inference process through graph neural networks. BAG(Cao et al., 2019b) further combines the bidirectional attention mechanism to perform graph reasoning in the entity-graph. However, limited sources of evidence hinder the performance of these methods. There are also studies using external knowledge to construct knowledge subgraphs with richer evidence. KagNet(Lin et al., 2019) extracts the knowledge subgraph from the structured knowledge base (ConceptNet(Speer et al., 2017)) according to the question and the candidate answer, and reason on the knowledge subgraph via a graph neural network. Graph Reasoning+XLNet(Lv et al., 2020) utilizes the evidence subgraphs extracted from ConceptNet and Wikipedia simultaneously to infer commonsense. Another approach is retrieving relevant knowledge from the knowledge base or text library and directly imparting knowledge to PTLMs(Pan et al., 2019; Ye et al., 2019; Zhang et al., 2018; Li et al., 2019). These methods cannot provide good coverage and interpretability. By contrast, our method combines multiple external knowledge bases to generate multi-source

subgraphs, which guarantees coverage. It also provides clearer interpretable reasoning through the attention between graph paths and question evidence.

2.2 Graph Neural Networks

Our work is also closely related to Graph Neural Networks (GNN), as we consider the GNN 1) contextually refines the concept vectors and 2) captures structural patterns of schema graphs for generalization. Graph Neural Networks (GNN) have been utilized widely in NLP. For example, the study in(Sun et al., 2019) utilizes Graph Convolutional Networks (GCN)(Kipf and Welling, 2017) to jointly extract entities and relations. A recently proposed extension of GCNs for relational graphs(Schlichtkrull et al., 2018) is applied to represent the relations between the evidence in commonsense reasoning tasks(Lin et al., 2019). Inspired by Relational Graph Convolutional Network (RGCN), a novel graph neural network MHGRNciteDBLP:conf/emnlp/FengCLWYR20 can leverage general knowledge by multi-hop reasoning over interpretable structures. Weighted Graph Convolutional Network (WGCN)(Shang et al., 2019) utilizes learnable relational scalar weights during GCN aggregation.

However, these approaches to handle multi-relational graphs suffer from over-parameterization and are difficult to learn effective representations of edges. A newly proposed novel Graph Convolutional based framework (CompGCN)(Vashishth et al., 2020) for multi-relational graphs leverages a variety of composition operations from knowledge graph embedding techniques to jointly embed nodes and relations in a graph. To process multi-relational graphs, CompGCN fits our framework perfectly. Thus CompGCN is used as the subgraph encoder in our method.

3 Overview

In this section, we start with the problem statement. Then we introduce the overall process of our proposed method.

3.1 Problem Statement

For a given commonsense question q and the corresponding answer set $\{a_i\}$ that contains K candidates, the task requires selecting the correct one from the candidate set. We assume that this task is a sorting problem: given a statement $s = a_i : q$ for question q and answer a_i in the candidate set, we calculate the matching score. Finally, the answer with the highest matching score is selected as the predicted answer.

3.2 Overall Framework

As described in Figure 2, our framework processes a pair of question and answer denoted as q and a. First, according to s we extract evidence from multiple external knowledge sources to form a subgraph (details in 4.1), to enrich the evidence needed for reasoning. Here, a piece of evidence is defined as one valuable relation or entity for a concrete question. In order to capture the particular contextual information within the subgraph, the relation-aware graph neural network module is adopted to encode the subgraph entities. To conduct bidirectional reasoning, the bidirectional path reasoning module further augments the inference by aggregating the entities and relations into paths. Then, an attention mechanism selectively aggregates important paths according to the statement vector s (represented by a model-agnostic language encoder) and calculate the graph vector g. The attention module is designed to provide explicit relational paths for model behavior interpretation. Finally, we concatenate the statement vector s and graph vector g, to calculate the matching score.

4 Proposed Method

In this section, we detail the four components of our model: subgraph construction, relation-aware graph neural network, bidirectional path reasoning, and the attention module.

4.1 Subgraph Construction

First, we select ConceptNet(Speer et al., 2017) as our knowledge base, a general-domain knowledge graph with millions of nodes and edges. ConceptNet can be seen as a set of triplets in the form of (h, r, t), such as (music, RelatedTo, singing), where h and t represent the head and tail concept in the concept set V, and r is a certain relation type from the pre-defined set R with 34 original relations.

Since not all knowledge is relevant, we distinguish which piece of evidence can help the model infer the correct answer. To this end, we only extract relevant evidence from the external knowledge base. Specifically, for each question and a candidate answer, we identify the question evidence e_q and the answer evidence e_a , by matching the to-

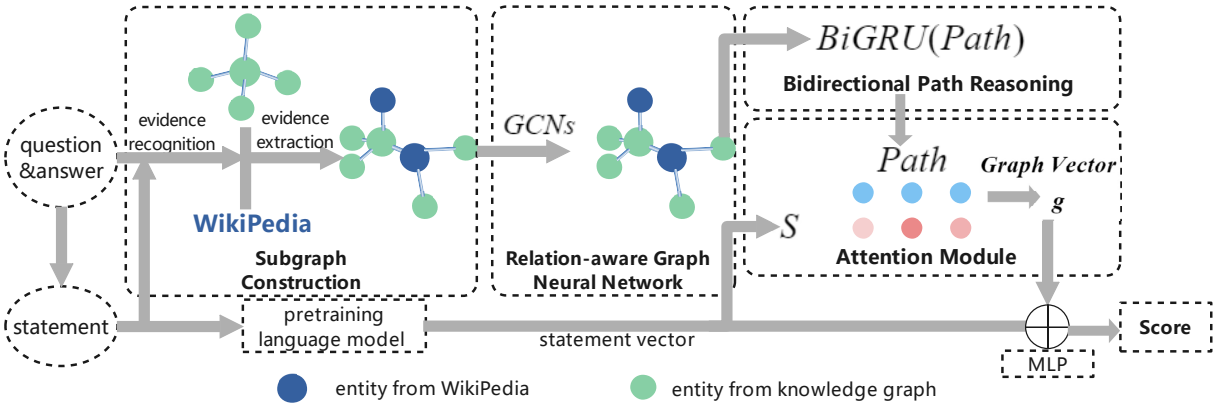

Figure 2: The overall workflow of the proposed framework.

kens in sentences and the concepts in the given ConceptNet graph. Then, we search for paths between the evidence pairs e_q and e_a, and merge these paths into a subgraph where nodes are entities and edges are the relations of ConceptNet. These paths will provide the necessary knowledge for inference. To extract the knowledge from WikiPedia, following the work in (Lv et al., 2020), we first use Spacy[2] to extract 107 million sentences from Wikipedia[3], and index the sentences by Elastic Search tools[4]. Then, in the given question and choices, we remove stopwords and concatenate the rest words as queries to search from the Elastic Search engine. This engine will match the queries against all Wikipedia sentences and rank them by the matching scores. Then we choose the top-K sentences (K is set as 3 in the experiment) and use Standford OpenIE[5] to extract the triples in them. We establish a relation matching mechanism between the relations in the triples that correspond to the 34 relations defined by ConceptNet. After replacing the relations, we insert all the triples into the subgraph to form a multi-source subgraph g. The vector representations of all nodes in the subgraph are obtained via TransE(Wang et al., 2014) pre-training. Representations of a newly added entity is randomly generated.

4.2 Relation-aware Graph Neural Network

Although we have obtained the evidence vector through pre-training, we need to further enrich the evidence by aggregating the contextual information in the subgraph. Since the graph structure provides potentially valuable contextual information for reasoning, we use Graph Convolutional Networks (GCNs)(Kipf and Welling, 2017) to update node vectors through the pooling features of neighboring nodes. Few GCNs are designed to process relations. RGCN(Schlichtkrull et al., 2018) can handle the *relations* but contains too many parameters to cause overfitting. We select CompGCN(Vashishth et al., 2020) as the network, since it can jointly embed relation vectors and node vectors with low resource consumption. Besides, CompGCN can avoid overfitting by replacing matrices with embedding vectors, and defining basis vectors only for the first layer, which reduces parameters. Specifically, all node embeddings $v_i \in V_g$ and relation embeddings $r_i \in R_g$ in the subgraph are first pre-trained or randomly generated $(h_i^0 = V_i)$. Then we extend the edges E by adding a reverse edge to all edges, and add an edge to itself for each node:

$$\begin{aligned}\bar{E} =& E \cup \left\{(v, u, r^{-1}) | (v, u, r) \ni E\right\} \cup \\ & \{(u, u, \perp) | u \ni V\}\end{aligned} \tag{1}$$

and $\bar{R} = R \cup R_{inv} \cup \perp$, where $R_{inv} = \left\{r^{-1} | r \in R\right\}$ denotes the inverse *relations* and $\perp$ indicates the self-loop. This allows the information in a directed edge to flow along both directions. After that, we update the node embeddings at the $(k+1)$-th layer by averaging the representations of their neighboring nodes (h_v^k) and edges (h_r^k):

$$\begin{aligned}h_v^{k+1} =& \sigma(W_{self}^k h_i^k + \\ & \sum_{(u,r) \in N(v)} \frac{1}{N(v)} W_{\lambda(r)}^k \phi(h_v^k, h_r^k))\end{aligned} \tag{2}$$

where ϕ indicates the *composition* of a neighboring node u with respect to its relation r as defined above. $\phi(h_v^k, h_r^k)$ is denoted by $(h_v^k) * (h_r^k)$. h_r is

[2]https://spacy.io/
[3]Wikipedia version enwiki-20190301
[4]https://www.elastic.co/
[5]https://github.com/philipperemy/Stanford-OpenIE-Python

obtained by a projection matrix W_{rel} of edge space and node space. $W_{\lambda(r)}$ is weighted by

$$\begin{cases} W_O, & r \in R. \\ W_I, & r \in R_{inv}. \\ W_S, & r = \perp (self-loop). \end{cases} \tag{3}$$

where W_O denotes the matrix of the original relations. W_I denotes the matrix of the inverse relations and W_S denotes the matrix of the self-loop relations. Then, the relation embeddings at the $(k+1)$-th layer are updated as:

$$h_r^{k+1} = W_{rel}^k h_r^k \tag{4}$$

4.3 Bidirectional Path Reasoning

This component aims to capture the path information between the question evidence and the answer evidence in the subgraph. When reasoning along the path, a reverse direction should entail different meanings of the relation. As shown in Figure 1, we want the model to capture the relation bias of "Capable Of" with respect to both directions "People→Capable Of→Singing" and "Singing→Capable Of→People", respectively. In this way, our model avoids information loss caused by the direction, which improves robustness of the model. Therefore, we encode the path by Bi-GRU. We denote the $c-th$ path formed between the $i-th$ question evidence and the $j-th$ answer evidence as follow:

$$Path_{i,j}^c = \left[e_i^{(q)}, r_0, v_0, r_1, ..., r_{n-1}, e_j^{(a)}\right] \tag{5}$$

We adopt Bi-GRU to encode these nodes, and concatenate the hidden state of the first and last cell.

$$P_{i,j,[c]} = GRU(Path_{i,j}^c) \tag{6}$$

4.4 Attention Module

Next, we jointly represent the learned path information and the text information conveyed in the question. Obviously, not all paths are meaningful: usually only a few paths play a decisive role in reasoning. Hence, we adopt the attention mechanism on the generated statement s and paths to selectively aggregate significant paths. Our model learns the parameter matrix W_1 for calculating path attention. The importance of the $c-th$ path between the $i-th$ question evidence and the $j-th$ answer evidence is denoted as $\alpha_{(i,j,c)}$.

$$\alpha_{(i,j,c)} = sW_1 P_{(i,j,[c])} \tag{7}$$

$$\bar{\alpha}_{(i,j,\cdot)} = softmax(\alpha_{(i,j,\cdot)}) \tag{8}$$

Then, we aggregate path $P_{(i,j,[c])}$ and evidence c_i, c_j as the vector representation g of the current subgraph:

$$g = \sum_{i,j}[\sum_c \bar{\alpha}_{(i,j,c)} P_{(i,j,[c])}; c_i; c_j], \tag{9}$$

where $[\cdot;\cdot]$ represents the concatenation of two vectors. The statement vector s in the above equation is obtained from a certain language encoder, which can either be a trainable sequence encoder like LSTM or features extracted from pre-trained universal language encoders like BERT/RoBERTa). To encode the text input into the distributed representation, the language encoder which is the pre-trained language model with a massive amount of corpus takes an input that is formalized as "[CLS]+Question+[SEP]+candidate answer." The score of the answer to the final question is obtained by inputting the concatenation of g and s into a fully connected layer: $score(q,a) = sigmoid(MLP([g;s]))$.

We define questions containing the word "not" as the *negative question*. Previous subgraph reasoning methods tend to choose the wrong answer when dealing with *negative question*. On the contrary, the pre-trained models do not suffer from this issue. Because they have learned the representation of the word "not", while the subgraph inference method has not. Hence, we set a hyperparameter η to control the weight between the statement s and the subgraph vector g when calculating the answer. And we add labels to all *negative question* manually in the dataset(train, development and test sets). We call this change the negation mode.

5 Experiment

5.1 Dataset

The CommonsenseQA dataset(Talmor et al., 2019) is a commonsense question answering dataset for multiple choice questions (Figure 3). It requires different types of commonsense knowledge to predict the correct answer and hence is quite challenging. To ensure that commonsense knowledge is used to answer the questions, each question contains an entity from ConceptNet. Each question corresponds to one correct answer and four interfering answers.

The official dataset has a total of 12,102 questions (official split: training/development/test

9,741/1,221/1,140). Since the prediction of the official test set can only be evaluated by the organizer within two weeks, we split the given training set into a new training and test set for evaluation. (new split: training/development/test 8,500/1,221/1,241)

5.2 Experimental Settings

In the best model on the development dataset, two CompGCN layers (100 dim and 50 dim respectively) and a bidirectional GRU (256 dim) are set. We use TransE (100 dimension) initialized by GloVe to pre-train the knowledge embedding. We use Roberta-LARGE, as a pre-trained sentence encoder to obtain the fixed characteristics of each pair of question and answer candidates. The input format for each choice is "[CLS]+Question+[SEP]+candidate answer." Totally, we get 5 confidence scores for all the choices. Then we adopt the softmax function to calculate the loss between the predictions and the ground truth. We adopt the cross-entropy loss as our loss function. We set the batch size as 4 and the learning rate $5e-6$. We use the Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

5.3 Baselines

We divide the comparison methods into two groups. The first group are pre-trained models without external knowledge, including BERT, XLNet, RoBERTa, ELECTRA. The above models are pre-trained on a large text corpus and then fine tuned on the training data. Prediction is conducted on the test set without extracting external knowledge for the current task. The second group consists of models with external knowledge, include KagNet(Lin et al., 2019), RoBERTa+IR, RoBERTa+KE, HyKAS(Ma et al., 2019), RoBERTa+KEDGN, XLNet+Graph Reasoning(Lv et al., 2020). These methods augment the inference by incorporating external knowledge. Among them, KagNet(Lin et al., 2019) constructs subgraphs from ConceptNet and uses the path-based attention mechanism for reasoning. XLNet+Graph Reasoning(Lv et al., 2020) combines different knowledge sources (ConceptNet, WikiPedia) to construct multiple subgraphs, and infer the answers through the graph network attention mechanism. RoBERT+KE and RoBERTa+IR adopt RoBERTa as the baseline and utilize the evidence from Wikipedia and search engine, respectively.

Model	NDev-Acc.(%)	NTest-Acc.(%)
BERT-Large	60.3	55.8
XLNet-large	74.8	62.1
RoBERTa-Large	72.8	69.1
ELECTRA-Large	77.9	72.1
BERT-Large-**RaB-PR**	63.3	60.8
XLNet-large-**RaB-PR**	75.1	71.9
RoBERTa-Large-**RaB-PR**	75.6	**73.7**
ELECTRA-Large-**RaB-PR**	**76.1**	72.3

Table 1: Comparisons with large pre-trained language model using the new split

Model	ODev-Acc.(%)	OTest-Acc.(%)
KagNet	64.4	58.9
RoBERTa+IR	78.9	72.1
HyKAS	-	73.2
RoBERTa+KE	-	73.3
RoBERTa+KEDGN	-	74.4
XLNet+Graph Reasoning	79.3	75.3
RaB-PR	**81.6**	**76.7**

Table 2: Comparison with official benchmark baseline methods using the official split

5.4 Performance Analysis

We conduct the experiments on our new splits to investigate whether our method can also work well on other universal language encoders (BERT, XLNet and ELECTRA). For the test set, Table 1 shows that our methods using fixed pre-trained language encoders outperform fine-tuning in all settings, achieving 60.8% with BERT-large, 71.9% with XLNet-large, 73.7% with RoBERTa-large, and 72.3% with ELECTRA-large. This indicates that the concept representations obtained from the our mothed had significant benefits and stable generality on CommonsenseQA, regardless of the language model encoder types.

Additionally, we evaluate our method on the official split. As shown in Table 2, we compare our model with the baseline methods reported on the paper and the leaderboard of commonsenseQA dataset. Methods based on RoBERTa and XLNet are much better than other baseline methods, demonstrating the ability of language models to utilize commonsense knowledge in an implicit way. Our method is similar to the architecture of KagNet. The difference is that our method focuses on the relations between the evidence pieces(details in section 4), which resulted a much bigger improvement(17.8%). Compared with XLNet+Graph Reasoning, our approach achieves a 1.4% absolute improvement, largely due to aggregating the multi-source knowledge. To conclude, our proposed

*Where on a **river** can you hold a cup upright to catch water on a sunny day?*
√***waterfall***, ×*bridge*, ×*valley*, ×*pebble*, ×*mountain*

*I'm crossing the **river**, my feet are wet but my body is dry, where am I?*
×*waterfall*, ×*bridge*, √***valley***, ×*bank*, ×*island*

*Where can I stand on a **river** to see water falling without getting wet?*
×*waterfall*, √***bridge***, ×*valley*, ×*stream*, ×*bottom*

Figure 3: A CommonsenseQA example (the bold green in the question represents the source concept; the blue font in the answer represents the target concepts sampled from ConceptNet, where there is a correct answer. The red and purple fonts represent the interfering concept sampled from ConceptNet.)

Model	NTest-Acc.(%)
RaB-PR	73.7
-w/o WikiPedia4.1	72.9
-Replace CompGCN w/GCN 4.2	70.2
-Replace CompGCN w/RGCN 4.2	71.1
-Replace BiGRU w/GRU 4.3	70.7
-w/o Attention 4.4	71.5
-w/o Negation Mode 4.4	72.2

Table 3: Ablation experiment using the new split

framework outperforms strong baselines including KagNet, HyKAS, XLNet+Graph Reasoning, achieving an absolute improvement of 1.4 points in accuracy on the test data, reaching a new state-of-the-art performance.

5.5 Ablation Study

In order to investigate the effectiveness of individual components in our method, we assess the impact of the model components on our new splits, shown in Table 3. Adding Wikipedia brings a 0.8% improvement as an additional source of knowledge, which proves that our model can effectively utilize evidence from multiple sources of knowledge. Compared with CompGCN, GCN and RGCN result in a 3.5% and 2.6% drop respectively. This confirms our initial hypothesis: 1.The relations contain rich information. 2.The way of encoding relations and entities affects the performance of the model. By replacing GRU with BiGRU, we can obtain a 3.0% gain. This proves the effectiveness of bidirectional path reasoning. Our attention mechanism is also critical, bringing a 2.2% benefit. The negation mode irons out 1.5% error cases. This proves that compared to traditional knowledge-aware methods, the negation mode can deal with negative questions better.

5.6 Interpretability

Our method can provide transparent interpretability through the bidirectional attention mechanism. Figure 5 shows an example from CommonsenseQA, where our model correctly answers the question and provides a reasoning path as evidence. In the example, our model first generates the paths between the evidence pieces based on the subgraph in Figure 1. Then the model follows the path with the highest attention score (dark red round in Figure 1) to reason for the correct answer. It is evident that the path is crucial for the inference process.

In order to further explore model interpretability, we randomly select 200 cases from the development dataset where each model predicts the correct answer. In these cases, we manually marked out the correct path. We consider a reasoning case correct when its reasoning path happens to have the highest attention score. We calculate the proportion of the correct reasoning case when using different GNNs, as shown in the Figure4. The results show that in 87.5% of the correct cases, our model chose a reasonable reasoning path. The much higher scores from models that use relations (CompGCN and RGCN) over models that do not (GCN) highlight the importance of learning *relations* which further improves the interpretability of reasoning.

5.7 Error Analysis

For qualitative analysis, we randomly select 50 error examples from the development dataset and conduct an analysis. There are two types of error cases:

1.Subgraph evidence error: 28% of the extracted subgraphs lack correct information, which makes it impossible to establish a path between the question evidence and the correct answer's evidence. 26%

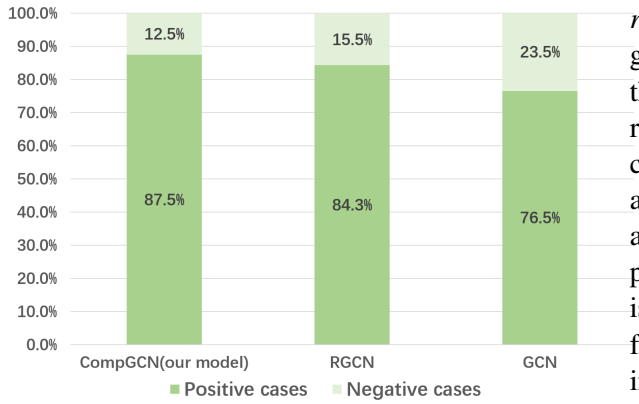

Figure 4: The comparison of model interpretability accuracy among GCNs.

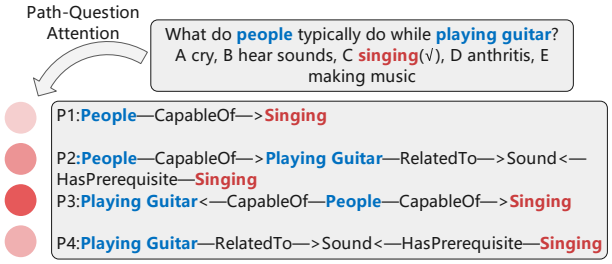

Figure 5: An example of interpreting the model behavior by the path-based attention mechanism. The blue entities correspond to evidence in the question. The red entities correspond to evidence in the answer. The red rounds on the left correspond to the attention scores.

cases of the error happen in the subgraph construction: the supplementary evidence imported from the WikiPedia is sometimes incorrect and contradicts with the correct relation evidence contained in the original knowledge graph. These errors could be fixed by further auditing and filtering of different evidence sources, which we leave for future work.

2.Path selection error:

We find that in 46% of the cases the model chooses the wrong reasoning path which has the highest attention score, even if the model has the correct evidence subgraph. This situation is usually due to the subjectivity of the problem or the lack of semantic information.

6 Conclusion

In this work, we propose a path-based reasoning framework that aims at obtaining high performance on CQA tasks while providing explicit relation paths for modeling behavior interpretation. Our framework introduces a relation-aware graph neural network to jointly embedding both nodes and *relations*, which enriches the evidence in the subgraph. The bidirectional reasoning module models the bidirectional reasoning process based on the rich evidence, and the attention mechanism efficiently augments the joint representations of path and question text. In addition, the attention mechanism enables our model to restore the reasoning path. Evaluation results show the proposed method is effective and achieves competitive results. In the future, we plan to introduce reinforcement learning and semantic information into the framework, and try to transfer the framework to multimodal commonsense reasoning tasks.

References

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019a. Question answering by reasoning across documents with graph convolutional networks. pages 2306–2317. Association for Computational Linguistics.

Yu Cao, Meng Fang, and Dacheng Tao. 2019b. BAG: bi-directional attention entity graph convolutional network for multi-hop reasoning question answering. pages 357–362. Association for Computational Linguistics.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems. *ACM SIGKDD Explorations Newsletter*, 19(2):25–35.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. OpenReview.net.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. pages 4171–4186. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. OpenReview.net.

Shiyang Li, Jianshu Chen, and Dian Yu. 2019. Teaching pretrained models with commonsense reasoning: A preliminary kb-based approach. *CoRR*, abs/1909.09743.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. pages 2829–2839. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019.

Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. 2020. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. pages 8449–8456. AAAI Press.

Kaixin Ma, Jonathan Francis, Quanyang Lu, Eric Nyberg, and Alessandro Oltramari. 2019. Towards generalizable neuro-symbolic systems for commonsense question answering. *CoRR*, abs/1910.14087.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James F. Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. pages 839–849. The Association for Computational Linguistics.

Simon Ostermann, Michael Roth, and Manfred Pinkal. 2019. Mcscript2.0: A machine comprehension corpus focused on script events and participants. pages 103–117. Association for Computational Linguistics.

Xiaoman Pan, Kai Sun, Dian Yu, Jianshu Chen, Heng Ji, Claire Cardie, and Dong Yu. 2019. Improving question answering with external knowledge. pages 27–37. Association for Computational Linguistics.

Hannah Rashkin, Maarten Sap, Emily Allaway, Noah A. Smith, and Yejin Choi. 2018. Event2mind: Commonsense inference on events, intents, and reactions. pages 463–473. Association for Computational Linguistics.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019. ATOMIC: an atlas of machine commonsense for if-then reasoning. pages 3027–3035. AAAI Press.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer.

Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. 2019. End-to-end structure-aware convolutional networks for knowledge base completion. pages 3060–3067. AAAI Press.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. pages 4444–4451. AAAI Press.

Changzhi Sun, Yeyun Gong, Yuanbin Wu, Ming Gong, Daxin Jiang, Man Lan, Shiliang Sun, and Nan Duan. 2019. Joint type inference on entities and relations via graph convolutional networks. pages 1361–1370. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. pages 4149–4158. Association for Computational Linguistics.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha P. Talukdar. 2020. Composition-based multi-relational graph convolutional networks. OpenReview.net.

Ruijie Wang, Meng Wang, Jun Liu, Weitong Chen, Michael Cochez, and Stefan Decker. 2019. Leveraging knowledge graph embeddings for natural language question answering. volume 11446 of *Lecture Notes in Computer Science*, pages 659–675. Springer.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. pages 1112–1119. AAAI Press.

Haocheng Wu, Zuohui Tian, Wei Wu, and Enhong Chen. 2017. An unsupervised approach for low-quality answer detection in community question-answering. volume 10178 of *Lecture Notes in Computer Science*, pages 85–101. Springer.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. pages 5754–5764.

Zhi-Xiu Ye, Qian Chen, Wen Wang, and Zhen-Hua Ling. 2019. Align, mask and select: A simple method for incorporating commonsense knowledge into language representation models. *CoRR*, abs/1908.06725.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. pages 93–104. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? pages 4791–4800. Association for Computational Linguistics.

Yuyu Zhang, Hanjun Dai, Kamil Toraman, and Le Song. 2018. Kgˆ2: Learning to reason science exam questions with contextual knowledge graph embeddings. *CoRR*, abs/1805.12393.