I Know What You Asked: Graph Path Learning using AMR for Commonsense Reasoning

Jungwoo Lim[*], **Dongsuk Oh**[*], **Yoonna Jang, Kisu Yang, Heuiseok Lim**[†]
Computer Science and Engineering, Korea University
Republic of Korea
{wjddn803,inow3555,morelychee,willow4,limhseok}@korea.ac.kr

Abstract

CommonsenseQA is a task in which a correct answer is predicted through commonsense reasoning with pre-defined knowledge. Most previous works have aimed to improve the performance with distributed representation without considering the process of predicting the answer from the semantic representation of the question. To shed light upon the semantic interpretation of the question, we propose an AMR-ConceptNet-Pruned (ACP) graph. The ACP graph is pruned from a full integrated graph encompassing Abstract Meaning Representation (AMR) graph generated from input questions and an external commonsense knowledge graph, ConceptNet (CN). Then the ACP graph is exploited to interpret the reasoning path as well as to predict the correct answer on the CommonsenseQA task. This paper presents the manner in which the commonsense reasoning process can be interpreted with the relations and concepts provided by the ACP graph. Moreover, ACP-based models are shown to outperform the baselines.

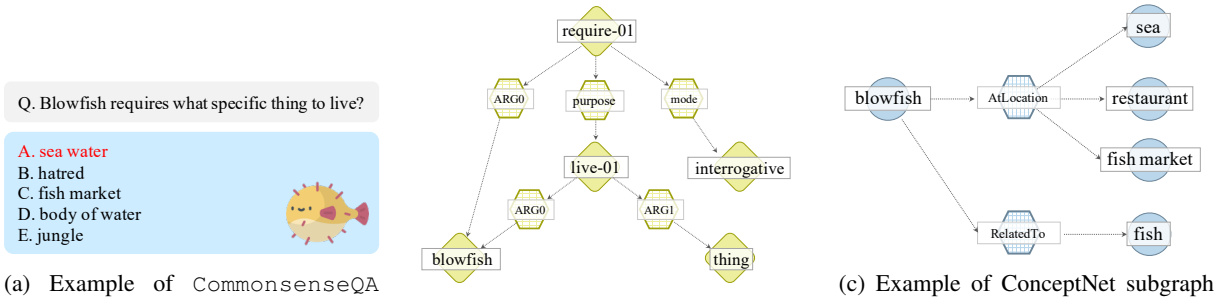
1 Introduction

Commonsense is the knowledge shared by the majority of people in society and acquired naturally in everyday life. Commonsense reasoning is the process of logical inference by using commonsense information. Commonsense to answer the questions that is "Blowfish requires what specific thing to live?" in Figure 1 is depicted as: "Blowfish is fish", "Fish lives in the water", and "Water includes seas and rivers." An enormous amount of pre-defined commonsense knowledge is available and people can make inferences using this commonsense such as in the following example: "Blowfish is fish." → "Fish lives in the water." → "Water includes seas and rivers." ⇒ "Blowfish lives in the sea." This chain of commonsense reasoning is naturally deduced by humans without substantial difficulty. Whereas people acquire commonsense in their lives, machines cannot learn this knowledge without any assistance. A large amount of external knowledge and several reasoning steps are required for machines to learn commonsense. In recent years, various datasets (Zellers et al., 2018; Sap et al., 2019; Zellers et al., 2019) have been constructed to enable machines to reason commonsense.

CommonsenseQA (Talmor et al., 2019) is one of the most widely researched datasets and is presented in Figure 1 (a). The studies of commonsense reasoning based on this dataset can be categorized into two mainstream approaches. The first approach uses pre-trained language models with distributed representations, which exhibit high performances on most Natural Language Processing (NLP) tasks. However, despite their high performance, these models must be trained with an excessive number of parameters and cannot explain the process of commonsense reasoning. The second approach is reasoning with a commonsense knowledge graph. The generally used commonsense knowledge graph is ConceptNet 5.5 (Speer et al., 2017), which includes parsed representation from Open Mind Commonsense (OMCS) and other different language sources such as WordNet (Bond and Foster, 2013) or DBPedia (Auer et al., 2007). In this approach, the subgraph of ConceptNet corresponding to the questions are transformed into

[*] Equal contribution
[†] Corresponding author

Proceedings of the 28th International Conference on Computational Linguistics, pages 2459–2471
Barcelona, Spain (Online), December 8-13, 2020



(a) Example of CommonsenseQA dataset

(b) Example of AMR graph of question

(c) Example of ConceptNet subgraph of question

Figure 1: Example of AMR and ConceptNet graphs of question.

node embeddings by the graph encoder. The candidate with the highest attention score is selected as an answer that is computed between the node embeddings and the word vectors from the language models. To learn the commonsense knowledge that is not observed or understood by the language models, relations from ConceptNet serve as a critical role in this method. The performance is improved by utilizing the relations that are not represented in the text; however, the interpretation of the question is still not enough.

Unlike CommonsenseQA, the most commonly used method of solving this problem is Knowledge-Based Question-Answering (KBQA) (Berant et al., 2013; Yih et al., 2014; Yih et al., 2015) employing semantic representations. As this method infers the answer with the logical structure of the question using the knowledge base, the question-answering process can be explained in a logical form. In our work, Abstract Meaning Representation (AMR) (Banarescu et al., 2013), which is one of the logical structure, is used to understand the overall reasoning process, from the question to the answer. AMR is a graph for meaning representation that symbolizes the meaning of sentences. AMR illustrates "who is doing what to whom" that is implied in a sentence with a graph. The components of these graphs are not the words, but rather the concepts and their relations. Each concept denotes an event or an entity, and each relation represents the semantic role of the concepts.

In this paper, we enable the language models to exploit the AMR graph to understand the logical structure of sentences. However, it is difficult to infer commonsense information with only an AMR graph, owing to its deficiency of commonsense knowledge of the given sentence. For example, in Figure 1 (b), the AMR graph indicates the path of the logical structure of the sentence "What the blowfish requires to live?" (require-01 → purpose → live-01 → ARG0 → blowfish); in other words, these paths from the single AMR graph lack the proficient information to predict the right answer. Therefore, for commonsense reasoning, dynamic interactions between the AMR graph and ConceptNet are inevitable to reach the correct answer.

Thus, we propose a new compact AMR graph expanded with the ConceptNet's commonsense relations with pruning, and it is called ACP graph. The proposed method can interpret the path from the question to the answer by performing commonsense reasoning within the connected graph, such as "Blowfish needs the sea to live." (require-01 → purpose → live-01 → ARG0 → blowfish → AtLocation → sea).

The contributions of our study are as follows.

- We introduce a new graph structure the ACP graph, which is pruned from a full integrated graph encompassing Abstract Meaning Representation (AMR) graph generated from input questions and an external commonsense knowledge graph, ConceptNet (CN) for commonsense reasoning. This structure is represented in a Levi graph (Gross et al., 2013) form to enable relation interpretations.
- We propose a graph-path reasoning framework, using which it is possible to explain the path from the question to the answer in a logical manner based on commonsense reasoning.
- Our path reasoning method exhibits a performance improvement over previous models.

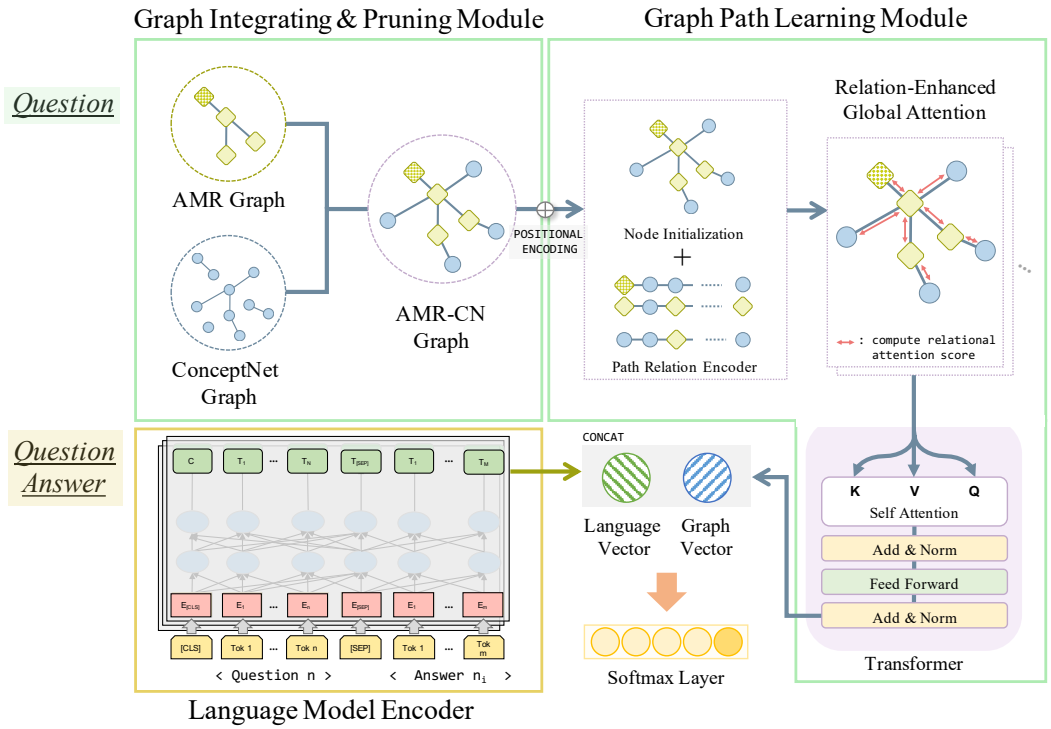


Figure 2: Overview of proposed method.

The remainder of this paper is organized as follows. In Section 2, we present the entire process of our method in detail. The experimental setup and results are explained in Section 3. A discussion of the proposed model is provided in Section 4, and Section 5 presents the conclusions. Appendix A provides related works including ConceptNet, previous works on commonsense reasoning, and AMR.

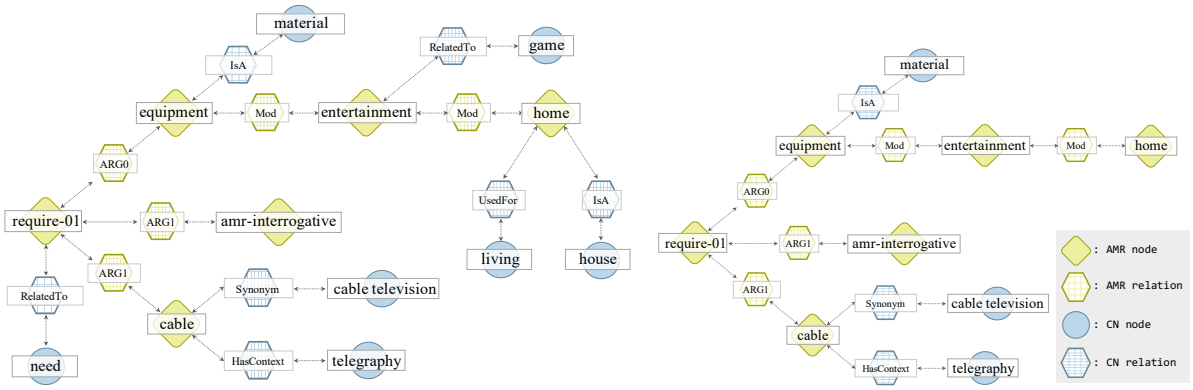
2 Proposed Method

We propose a commonsense reasoning framework that uses a commonsense knowledge base on the basis of the AMR logic structure. Our framework consists of the AMR graph integrating and pruning module, language model encoder, and graph path learning module[§]. As illustrated in Figure 2, we first generate the AMR graph from every question in the `CommonsenseQA` dataset and integrate all the nodes of AMR with ConceptNet graphs.

As this AMR-ConceptNet full graph also includes some irrelevant relations to the question, interpreting questions can be guided in the wrong way. For this reason, we suggest a new method, the ACP graph, pruned according to the relation type. Thereafter, the graph path learning module takes the pruned graph as an input and computes the attention score of each path by using the Graph Transformer (Cai and Lam, 2019) which results in the whole graph vector. The graph vector is finally fed into the Transformer (Vaswani et al., 2017) to model the interactions between the AMR and ConceptNet graph and transforms to the final graph representation. Meanwhile, the question and candidate answer from the dataset are passed through the language model encoder, producing the language vector. The concatenation of the language and graph vectors turns out to the final representation that is used to predict the correct answer.

In contrast to other models mentioned in Talmor (2019), which cannot provide interpretable reasons for predicting the correct answer from the question, our proposed method produces the reasoning paths that make the model transparent and interpretable. That is, the reasoning paths that have high attention weights from the graph encoder possess potentially accurate information for reasoning. These reasoning paths are depicted in Figure 5.

[§]Code available at `https://github.com/dlawjddn803`



(a) AMR-CN graph – full

(b) AMR-CN graph – pruned

Figure 3: Two methods of integrating AMR graph with ConceptNet graph. Method (a) incorporates all of the nodes from the AMR graph with the ConceptNet graph. In contrast, method (b) removes the ConceptNet nodes that is not connected to the AMR nodes which have `ARG0` and `ARG1` relations. For instance, we generate an AMR graph from the sentence "`What home entertainment equipment requires cable?`", and we find every ConceptNet relation that includes all the AMR nodes such as `(cable-HasContext-telegraphy)`, `(require01-RelatedTo-need)`, and `(home-UsedFor-living)`. Then remove the ConceptNet nodes that are not connected to the AMR nodes which have `ARG0` or `ARG1` relations. In this example, the deleted nodes are `living`, `house`, and `game`. In addition, as the `require-01` is frame node which does not need to expand, `need` is also removed from the graph.

2.1 Graph Integrating and Pruning

As each word plays a certain role as a predicate or an argument in a sentence, the concepts of the AMR graph also carry semantic meanings in the graph structure. Hence, the AMR graph is capable of interpreting the questions as paths, semantically . Owing to these advantages of the graph structure and preserved semantic interpretation, we use the AMR graph for extracting commonsense knowledge graph. To generate an AMR graph from the raw text, we use the pre-trained model of Zhang (2019), which is an attention-based model that treats AMR parsing as sequence-to-graph transduction. Though most of the AMR graphs generated from the model properly, they might have some inevitable errors in the type of relations or concept.

We suggest effective AMR expansion and pruning rules for commonsense reasoning. We expand the AMR graph on all nodes with the ConceptNet as illustrated in Figure 3 (a), and prune the nodes that have edges known as `ARG0` and `ARG1` with ConceptNet. Considering that `ARG0` and `ARG1` are the top two frequent relations among any other relations as shown in Table 1, we prune the full AMR-CN graph into a more compact graph that only contains `ARG0` and `ARG1` relations, which is called ACP graph. This

Relation	Ntrain	Ndev	Ntest
ARG0	**17300 (22.70%)**	**2547 (22.73%)**	**2477 (23.09%)**
ARG1	**24673 (32.38%)**	**3566 (31.83%)**	**3521 (32.82%)**
ARG2	6001 (7.88%)	864 (7.71%)	829 (7.73%)
ARG3	286 (0.38%)	37 (0.33%)	51 (0.48%)
ARG4	587 (0.77%)	92 (0.82%)	59 (0.55%)
Total relations	76203	11204	10727

Table 1: Statistics of core roles in `CommonsenseQA` AMR graph. We split the given training set into the new training and test sets randomly to conduct diverse experiments with efficiency. The new training, development, and test sets included 8,500, 1,221, and 1,241 examples, respectively.

procedure prevents the graph from discovering a tremendous number of paths iteratively. As described in Appendix A, since the frame node is defined as a central point in the AMR graph like `require-01` in Figure 3, combining other ConceptNet relations with the root node may distract the process of path reasoning. Also, the frame node's specific meaning additionally annotated by the number like "`-01`" at the end of the word is different from the meaning in ConceptNet's node even though it has identical letters. For example, the specific meaning of the frame node "`play-11`" is "play/perform music" defined in Propbank frameset while ConceptNet's node "`play`" includes more diverse meanings such as "engage in an activity like game"or "bet or wager". Therefore, we remove the ConceptNet relations and nodes connected to the frame node. The proposed method is depicted in Figure 3 (b).

The graph G = (V, E) expresses fixed set of nodes V, and relation edges E. Following this notation, the ACP graph is defined as follows:

$$\mathcal{G}_{ACP} = (\{\mathcal{V}_{amr} \cup \mathcal{V}_{cn}^{amr^{arg}}\}, \{\mathcal{E}_{amr} \cup \mathcal{E}_{cn}^{amr^{arg}}\}) \tag{1}$$

The ACP graph expressed in equation (1) is the union set of the AMR and the subgraph of ConceptNet that contains AMR concepts that are connected to `ARG0` and `ARG1`, respectively. The AMR graph is denoted as $\mathcal{G}_{AMR} = \{\mathcal{V}_{amr}, \mathcal{E}_{amr}\}$. The subgraph of ConceptNet matched with the concepts that are connected to `ARG0` and `ARG1` is defined as $\mathcal{G}_{CN}^{AMR^{arg}} = \{\mathcal{V}_{cn}^{amr^{arg}}, \mathcal{E}_{cn}^{amr^{arg}}\}$

2.2 Language Encoder and Graph Path Learning Module

The proposed method performs commonsense reasoning over the ACP graph and predicts the correct answer with the corresponding inference. Our model receives two types of inputs, which are text and graph, and converts semantic representation to distributed representation. 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`." Given the ACP graph from the graph integrating and pruning module, the graph path learning module initializes the concept node vectors as the sum of the concept embedding using GloVe (Pennington et al., 2014) and absolute position embedding. Inspired by the works of Cai (2019), we modify the graph transformer to make the model reason over the relation paths of the ACP graph. To let the model recognize the explicit graph paths, we first encode the relation between two concepts into a distributed representation using the relation encoder. The relation encoder identifies the shortest path between two concepts and represents the sequence as a relation vector by employing recurrent neural networks with a Gated Recurrent Unit (GRU) (Cho et al., 2014). The equation for the represented relation is expressed as follows:

$$\overrightarrow{p_t} = \mathrm{GRU}_f(\overrightarrow{p_{t-1}}, sp_t), \overleftarrow{p_t} = \mathrm{GRU}_g(\overleftarrow{p_{t+1}}, sp_t) \tag{2}$$

where sp_t indicates the shortest path of the relation between two nodes. The final relation encoding r_{ij} between concepts i and j is the concatenation of the final hidden states from the forward and backward GRU networks, which are presented in the equation (3).

$$r_{ij} = [\overrightarrow{p_n}; \overleftarrow{p_0}] \tag{3}$$

To inject this relation information into the concept representation, we follow the idea of relative position embedding (Shaw et al., 2018; Salton et al., 2017), which introduces the attention score method based on both the concept representations and their relation representation. To compute the attention score, we split the relation vector r_{ij} passed from the linear layer into forward relation encoding $r_{i \to j}$ and backward relation encoding $r_{j \to i}$, as follows:

$$[r_{i \to j}; r_{j \to i}] = W_r r_{ij} \tag{4}$$

where W_r is the parameter matrix. This split renders the model consider bidirectionality of the path.

Thereafter, we compute the attention score considering the concepts and their relations. Note that c_i and c_j are the concept embedding. The equation is presented below:

$$\begin{aligned} s_{ij} &= f(c_i, c_j, r_{ij}) \\ &= (c_i + r_{i\to j})W_q^\top W_k(c_j + r_{j\to i}) \\ &= c_i W_q^\top W_k c_j + c_i W_q^\top W_k r_{j\to i} \\ &+ r_{i\to j} W_q^\top W_k c_j + r_{i\to j} W_q^\top W_k r_{j\to i} \end{aligned} \tag{5}$$

The first term in the last line of equation (5) is the original term in the vanilla attention mechanism, which includes the pure contents of the concept. The second and third terms capture the relation bias with respect to the source and target, respectively. The final term represents the universal relation bias. As a result, the computed attention score updates the concept embedding while maintaining fully-connected communication (Cai and Lam, 2019). Therefore, concept–relation interactions can be injected into the concept node vector. The resulted concept representations are summed into the whole graph vector and fed into the Transformer Layers to model the interaction between AMR and ConceptNet concept representation. The major advantage of this relation-enhanced attention mechanism is that it provides a fully connected view of input graphs by making use of the relation multi-head attention mechanisms. Since we integrate two different concept types from the AMR graph and ConceptNet into a single graph, the model globally recognizes which path has high relevance to the question during the interpretation. After obtaining the language and graph vectors, the model concatenates the two vectors, feed these into the Softmax layer, and selects the correct answer.

3 Experiments

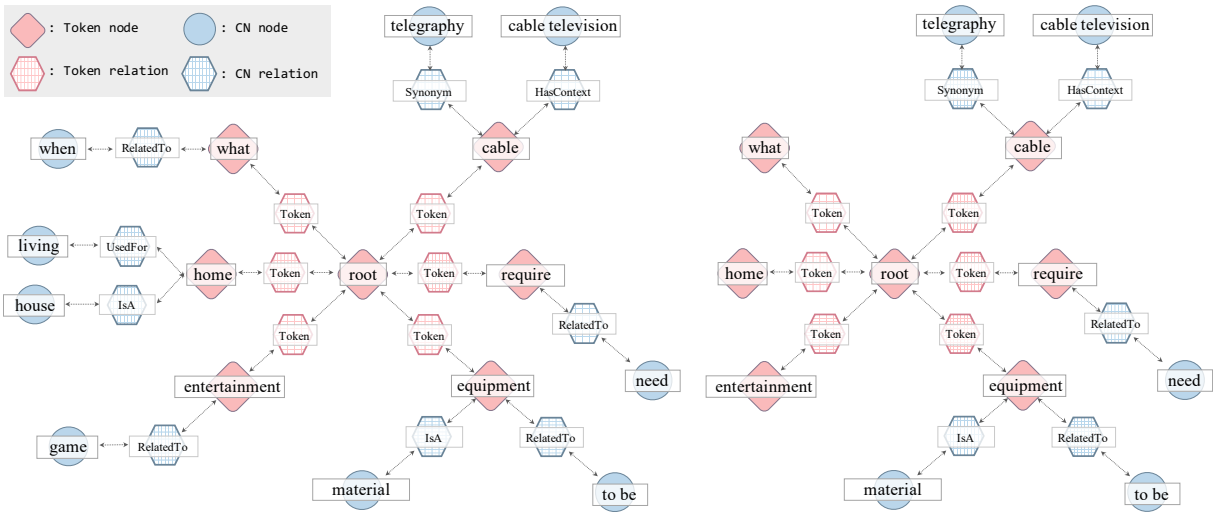
To show the effectiveness of representing a question using the proposed ACP graph in commonsense reasoning, we conduct four different experiments. We first compare the ACP with the ACF graph, which is expanded on all the concepts of the AMR graph, and with the graph that only utilizes the ConceptNet. In addition, we apply our model to three language models that have different encoder structures, showing performance enhancement as shown in Table 4. Moreover, we investigate the efficacy of the proposed method on the extended versions of the BERT-base model such as the BERT-large-cased or post-trained BERT model with OMCS data. Finally, we show the performance of our model with official test set.

3.1 Data and Experimental Setup

The `CommonsenseQA` dataset consists of 12,102 (v1.11) natural language questions and each question has five candidate answers provided by Talmor et al (2019). As the prediction on the official test set can be evaluated only through the organizers by two weeks, we divide the official training set for the experiment efficiency. We split the given training set into the new training and test sets. The new training, development, and test sets included 8,500, 1,221, and 1,241 examples, respectively. We use RTX8000 for training our model. The parameters for the graph path learning model are identical to the work in Cai (2019)'s model.

3.2 Experimental Results

Commonsense reasoning. To demonstrate that our ACP graph is more effective than other graph features, we conduct experiments on diverse graph features that include not only the ACP graph but also the ACF graph. The ACF graph is an integrated graph, with the AMR graph and the ConceptNet matched with all concepts from the AMR graph. Furthermore, we run experiments only using ConceptNet (CN) in two manners. The one is a method that uses the ConceptNet graph that corresponds to all the question tokens separated by the spaces of the sentence as depicted in Figure 4 (a). As the tokens from the question are not connected initially, it may prevent our graph path learning module from reasoning over the CN graph due to the disconnection between the concept nodes. Therefore, we connect all of the tokens from the question to the `root` node to let our model perform effectively on commonsense reasoning. The other is a method that employed the pruned ConceptNet graph using the the logic of AMR graph



(a) ConceptNet – full

(b) ConceptNet – pruned

Figure 4: Two methods of ConceptNet expansion matched with tokens. The token nodes are all the words in the question

as decribed in Figure 4 (b). The pruned ConceptNet graph includes the subgraph of the ConceptNet matched with tokens that are connected to `ARG0` and `ARG1` of the AMR. For example, as the concept nodes `require`, `equipment`, and `cable` in Figure 3 (a) are connected with the relation of `ARG0` and `ARG1`, only those concept nodes have the ConceptNet relations. Similar to the first method, the tokens from the question are linked with `token` relation to the `root` node. Note that these two methods do not explicitly make use of the AMR graph concepts.

ACF graph is depicted in Figure 3 (a) and is expressed as follows.

$$\mathcal{G}_{ACF} = (\{\mathcal{V}_{amr} \cup \mathcal{V}_{cn}^{amr}\}, \{\mathcal{E}_{amr} \cup \mathcal{E}_{cn}^{amr}\}) \tag{6}$$

Note that the AMR graph is denoted as $\mathcal{G}_{AMR} = \{\mathcal{V}_{amr}, \mathcal{E}_{amr}\}$ and the subgraph of ConceptNet matched with the ACF graph is denoted as $\mathcal{G}_{CN}^{AMR} = \{\mathcal{V}_{cn}^{amr}, \mathcal{E}_{cn}^{amr}\}$.

The (1) CN full graph (CF) and (2) CN pruned graph (CP) are illustrated in Figure 4 and defined as follows, respectively:

$$\mathcal{G}_{CF} = (\{\mathcal{V}_{token} \cup \mathcal{V}_{cn}^{token}\} \cup \{\texttt{root}\}, \{\mathcal{E}_{cn}^{token} \cup \{\texttt{token}\}\}) \tag{7}$$

$$\mathcal{G}_{CP} = (\{\mathcal{V}_{token} \cup \mathcal{V}_{cn}^{amr^{arg}}\} \cup \{\texttt{root}\}, \{\mathcal{E}_{cn}^{amr^{arg}} \cup \{\texttt{token}\}\}) \tag{8}$$

The ConceptNet graph is denoted as $\mathcal{G}_{CN} = \{\mathcal{V}_{cn}, \mathcal{E}_{cn}\}$ and the subgraph of ConceptNet matched with the question token is denoted as $\mathcal{G}_{CN}^{token} = \{\mathcal{V}_{cn}^{token}, \mathcal{E}_{cn}^{token}\}$. For the CN pruned graph, the notation is as follows: the subgraph of ConceptNet matched with the tokens connected to `ARG0` and `ARG1` is defined as $\mathcal{G}_{CN}^{AMR^{arg}} = \{\mathcal{V}_{cn}^{amr^{arg}}, \mathcal{E}_{cn}^{amr^{arg}}\}$. In addition, $\mathcal{V}_{token}$ denotes the token of the sentence that is separated by a space.

As indicated in Table 2, whereas the BERT fine-tuning only score 51.59% in terms of accuracy, the models with the AMR graph or ConceptNet exceed this result, scoring over 52%. Interestingly, the ACP graph achieves the best score among all other graph types. These results demonstrate that the ACF graph and other CF, CP graphs consider all possible paths which are unnecessary and obtain insufficient performance. In other words, the ACP graph enables the Graph Path Learning Module to find reasonable paths efficiently by ignoring the irrelevant paths. Since the ACP graph provides the best results, other experiments are conducted with the ACP graph feature.

Language Encoder	Graph type	Ndev-Acc.(%)	Ntest-Acc.(%)
BERT-base-cased	-	51.81	51.59
	AMR - original	52.82	52.78
	CN – full (*CF*)	53.80	53.10
	CN – pruned (*CP*)	52.61	52.53
	AMR-CN – full (*ACF*)	52.98	52.94
	AMR-CN – pruned (*ACP*)	**53.97**	**53.58**

Table 2: Experiments on diverse graph types.

Extensions of BERT. We also demonstrate the effects of the proposed method on improved BERT models that are post-trained BERT-base-cased with OMCS and BERT-large. Previous studies mostly addressed the `CommonsenseQA` utilizing a post-training approach with OMCS data, which is a freely available crowd-sourced knowledge base of natural language statements regarding the world. Because of the high performance on the ACP graph feature in Table 2, we illustrate the effect on the performance of the ACP graph with respect to the different version of BERT. As indicated in Table 3, our method combined with the post-trained BERT on OMCS achieved 54.31% in the new test set. Moreover, the BERT-large model with our method outperformed the BERT-large fine-tuning model, obtained 58.98% in the new test set.

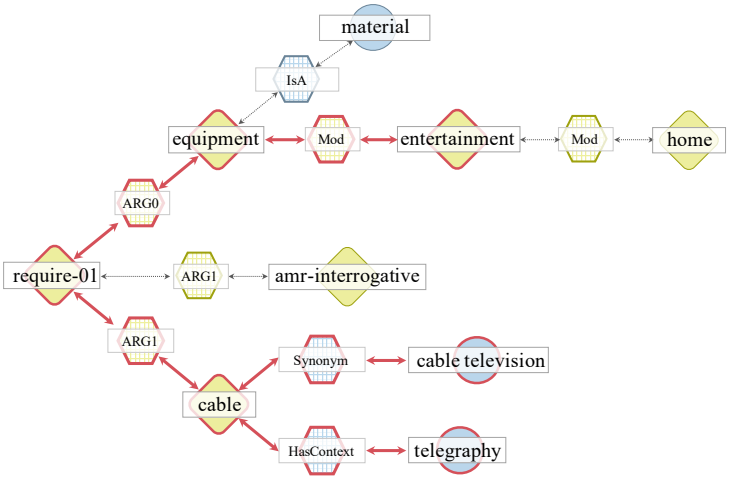
Language model	Ndev-Acc.(%)	Ntest-Acc.(%)
BERT post-trained w/ OMCS	52.13	52.08
BERT-large-cased	57.16	56.24
BERT post-trained w/ OMCS & AMR-CN – pruned(*ACP*)	**54.79**	**54.31**
BERT-large-cased w/ AMR-CN – pruned(*ACP*)	**59.37**	**58.98**

Table 3: Experiments on other BERT models.

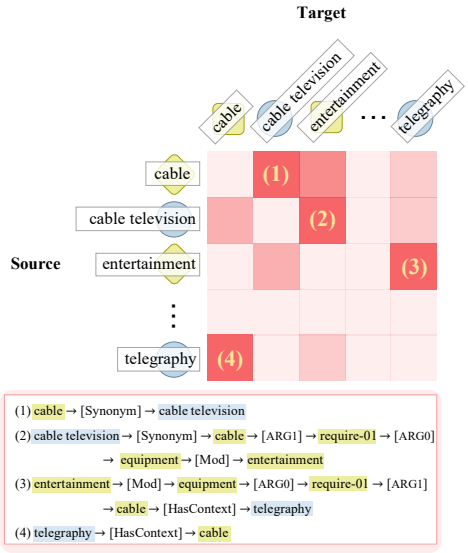
Comparison on different language models. Table 4 presents the results of the comparison experiments on different language models with varying transformer encoder structures. Input of the language model is formalized as "`[CLS]+Question+[SEP]+candidate answer`." All of the language models that use our method outperformed their own fine-tuned score, achieving 53.58% with BERT-base, 60.35% with XLNet-base (Yang et al., 2019), 51.08% with ALBERT-base (Lan et al., 2019), and 70.91% with ELECTRA-base (Clark et al., 2020) on our new test set. This implies that the concept representations obtained from the our ACP graph had significant effects and stable generality on `CommonsenseQA`, regardless of the language model encoder types.

Language Encoder	Ndev-Acc.(%)	Ntest-Acc.(%)
BERT-base-cased	51.81	51.59
XLNet-base-cased	57.98	57.05
ALBERT-base	50.12	49.22
ELECTRA-base	71.25	70.19
BERT-base-cased w/ AMR-CN – pruned (*ACP*)	**53.97**	**53.58**
XLNet-base-cased w/ AMR-CN – pruned (*ACP*)	**61.01**	**60.35**
ALBERT-base w/ AMR-CN – pruned (*ACP*)	**51.51**	**51.08**
ELECTRA-base w/ AMR-CN – pruned (*ACP*)	**71.99**	**70.91**

Table 4: Experiments on different language models.



(a) AMR-CN extended graph with high attention weight relations



(b) Heatmap of path attention from source to target words. The details of each path are presented below the heatmap

Figure 5: Case study of the question "`What home entertainment equipment requires cable?`" and candidate answers `(a) radio shack`, `(b) substation`, `(c) cabinet`, `(d) television`, `(e) desk`. Figure 5 (a) presents the entire AMR-CN graph marked with the red color for high attention weight paths in BERT-base-cased w/ AMR-CN–pruned graph (ACP). Figure 5 (b) displays the heatmap of the attention weights with respect to the source and target tokens in the path. The details of the paths are also provided.

Experiment on official test set As the ELECTRA-base model with the ACP graph shows the highest performance on the new test set, we conduct the experiments on the official test set (1140 examples) with the official training set utilizing ELECTRA-large. For official test set, our model achieve 75.43% accuracy.

Models	Odev-Acc.(%)	Otest-Acc.(%)
KagNet (Lin et al., 2019)†	64.46	58.90
HyKAS (Ma et al., 2019)†	80.10	73.20
XLNet + Graph Reasoning (Lv et al., 2019)†	79.30	75.30
ELECTRA-large w/ AMR-CN – pruned (*ACP*)	**82.15**	**75.43**

Table 5: Experiment with ELECTRA with full training set on the official test set. † denotes the results of the models that use ConceptNet, taken from the official commonsenseQA leaderboard in April 2020.

4 Discussion

4.1 Error Analysis

In some cases of failure, our model exhibits two problems as follows:

- **Difficulty in discriminating hard distractors**: All candidate answers from the `CommonsenseQA` possess a hard distractor, which shares the same relation with the question. When the hard distractor exists in the ACP graph in the path learning module, it also uses the paths of the distractor, instead of the those of correct answer. This may make the model confused as it considers the distractor as the correct answer.

- **AMR graph generation error**: Since the AMR graph is generated from the pre-trained model, our model is at risk of using an incorrect AMR graph. An incorrectly produced AMR graph may lead the model to incorrect interpretation and distortion in the wrong direction during the path reasoning process. For example, the AMR graph generated from the question "`What can help you with an illness?`" is described below.

```
(vv1 / possible
      :ARG1 (vv2 / help-01
            :ARG0 (vv3 / amr-unknown)))
```

  As the concept node `illness` disappeared while generating the graph, our model may not have enough information for extracting the subgraph from ConceptNet.

4.2 Case Study

The red edges in Figure 5 present the paths that have high attention weight for the question "`What home entertainment equipment requires cable?`" In Figure 5 (b), the top four paths with high attention weights are described. As opposed to predicting the answers simply with the ConceptNet graph connected to the question, we allow our model to learn relevant paths inherent in the ACP graph. That is, our graph path learning module with ACP graph is capable of commonsense reasoning exploring the paths.

5 Conclusions and Future Works

We introduce a new commonsense reasoning method, using the proposed ACP graph. This method outperformed the model that simply learns the ConceptNet graph. Furthermore, our method can explain the answer-inference process by interpreting the logical structure of the sentences within commonsense reasoning process. Models that applied our method exhibit higher performance compared to the previous models. However, certain problems still remain. Though the relations `ARG0` and `ARG1` occupy most of the core roles in the AMR graph, it is still arguable that the other choice of relations may lead to better results. Therefore, we will show the experimental results according to the different pruning rules on the `CommonsenseQA` task in the future. Also, we plan to develop an end-to-end learning model that incorporates the AMR generation model and the question-answering model to reduce the error propagation from the AMR generation.

6 Acknowledgement

This work was supported by Institute for Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2020-0-00368, A Neural-Symbolic Model for Knowledge Acquisition and Inference Techniques). Also, this research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2020-2018-0-01405) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation)

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse*, pages 178–186.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Francis Bond and Ryan Foster. 2013. Linking and extending an open multilingual wordnet. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1352–1362.

Claire Bonial, Jena Hwang, Julia Bonn, Kathryn Conger, Olga Babko-Malaya, and Martha Palmer. 2012. English propbank annotation guidelines. *Center for Computational Language and Education Research Institute of Cognitive Science University of Colorado at Boulder*, 48.

Claire Bonial, Lucia Donatelli, Stephanie Lukin, Stephen Tratz, Ron Artstein, David Traum, and Clare Voss. 2019. Augmenting abstract meaning representation for human-robot dialogue. In *Proceedings of the First International Workshop on Designing Meaning Representations*, pages 199–210.

Deng Cai and Wai Lam. 2019. Graph transformer for graph-to-sequence learning. *arXiv preprint arXiv:1911.07470*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

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*.

Sahil Garg, Aram Galstyan, Ulf Hermjakob, and Daniel Marcu. 2016. Extracting biomolecular interactions using semantic parsing of biomedical text. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Jonathan L Gross, Jay Yellen, and Ping Zhang. 2013. *Handbook of graph theory*. CRC press.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions of the Association for Computational Linguistics*, 7:297–312.

Fuad Issa, Marco Damonte, Shay B Cohen, Xiaohui Yan, and Yi Chang. 2018. Abstract meaning representation for paraphrase detection. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 442–452.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Kexin Liao, Logan Lebanoff, and Fei Liu. 2018. Abstract meaning representation for multi-document summarization. *arXiv preprint arXiv:1806.05655*.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2822–2832.

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. *arXiv preprint arXiv:1907.11692*.

Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. 2019. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. *arXiv preprint arXiv:1909.05311*.

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

Arindam Mitra and Chitta Baral. 2016. Addressing a question answering challenge by combining statistical methods with inductive rule learning and reasoning. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Giancarlo Salton, Robert Ross, and John Kelleher. 2017. Attentive language models. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 441–450.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. 2019. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of NAACL-HLT*, pages 4149–4158.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Andreas Vlachos et al. 2018. Guided neural language generation for abstractive summarization using abstract meaning representation. *arXiv preprint arXiv:1808.09160*.

Yanshan Wang, Sijia Liu, Majid Rastegar-Mojarad, Liwei Wang, Feichen Shen, Fei Liu, and Hongfang Liu. 2017. Dependency and amr embeddings for drug-drug interaction extraction from biomedical literature. In *Proceedings of the 8th acm international conference on bioinformatics, computational biology, and health informatics*, pages 36–43.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Wen-tau Yih, Xiaodong He, and Christopher Meek. 2014. Semantic parsing for single-relation question answering. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 643–648.

Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. *arXiv preprint arXiv:1808.05326*.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6720–6731.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019. Amr parsing as sequence-to-graph transduction. *arXiv preprint arXiv:1905.08704*.

Appendix A. Related Works

We briefly describe the structures and properties of ConceptNet, which consists of commonsense relations. Furthermore, we review previous studies and AMR, which is an essential part of our model.

ConceptNet. In ConceptNet (Speer et al., 2017), real-world assertions are represented as two nodes and directed edges, which denote certain concepts and their relations, respectively. The nodes represent words or phrases from natural language sentences. The edges represent the relations between nodes, and they contain lexical as well as commonsense relation information. As ConceptNet is created by collecting data from various types of knowledge bases, nodes of different types also exist. Each node represents a slightly different meaning considering its role in the sentence. For example, the word "`person`" can be found in the concept of "`person/n`," which is analyzed as a noun with a POS tagger, and with more detailed semantic information, it can be identified as "`person/n/wn/body`." This information makes possible the detailed extraction of knowledge that considers the purpose of each sentence. Meanwhile, one or more edges may be defined between two nodes. For example, the edge between the nodes "`person`" and "`eat`" can be defined independently as "`CapableOf`" and "`Desires`." Various concepts and their relations are defined as nodes and edges in ConceptNet, considering the ambiguity in the sentences.

Commonsense reasoning. Commonsense reasoning is the process of logical inference by using commonsense information. In `CommonsenseQA`* task, the fine-tuning approach with pre-trained language representations makes use of external commonsense knowledge. There are two means of exploiting external knowledge. The first** is the method that post-trained with some commonsense sentence corpus. It then performs fine-tuning with evidence derived from questions and answers. The second method (Lv et al., 2019; Lin et al., 2019) is to encode commonsense knowledge graphs and train with language models. The language models that have exhibited high performance in this method are BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), which use bidirectional transformer encoders. They also include XLNet (Yang et al., 2019), which is based on autoregressive language modeling, ALBERT (Lan et al., 2019), which adopts cross-layer parameter sharing and factorized embedding parameterization and ELECTRA (Clark et al., 2020) that is pre-trained with Replaced Token Detection (RTD) task.

AMR. AMR (Banarescu et al., 2013) represents the relations between concept nodes using the PropBank frameset and vocabularies from the sentences. The edges between two or more concept nodes or the argument nodes are relations. AMR represents semantic roles such as core and numbered roles, and uses more than 100 semantic relations, including negation, conjunction, command, and wikification. In PropBank (Bonial et al., 2012), the semantic roles are labeled in the form of `ARG0`$\sim$`4` and `ARGM`. In general, `ARG0` denotes the agent of the verb, `ARG1` is the patient, `ARG2` means the instrument, benefactive, or attribute, `ARG3` is interpreted as the starting point, benefactive, or attribute, and `ARG4` represents the ending point. The root node serves as the central point of the representation and is called frame node. Thereafter, other concept nodes are sequentially combined according to the semantic relations. AMR consists of concept nodes in a single graph that is traversable to all nodes, similar to a parse tree. However, unlike the parse tree, which represents the explicit structure of sentences, AMR aims to describe the conceptual and semantic structure. That is, if the semantic meanings of explicitly different sentences are the same, they can be represented by the same AMR graph. For example, the two sentences "`The boy is a hard worker`" and "`The boy works hard`" are represented by the same PENMAN graph, namely `(w / work-01 :ARG0 (b / boy) :manner (h / hard))`. The data constructed to generate and evaluate these representations are AMR 2.0 (LDC2017T10) and AMR 1.0 (LDC2014T12). The model with the highest performance on these data was presented by Zhang et al. (2019), using BERT. Various NLP fields have exploited AMR, such as sentence generation (Cai and Lam, 2019; Guo et al., 2019), summarization (Vlachos and others, 2018; Liao et al., 2018), question and answering (Mitra and Baral, 2016), dialogue systems (Bonial et al., 2019), paraphrase detection (Issa et al., 2018), and biomedical text mining (Wang et al., 2017; Garg et al., 2016).

*https://www.tau-nlp.org/csqa-leaderboard

**ttps://drive.google.com/file/d/1sGJBV38aG706EAR75F7LYwCqci9ocG9i/view, https://gist.github.com/commonsensepretraining/507aefddcd00f891c83ebf6936df15e8