Dynamic Relevance Graph Network for Knowledge-Aware Question Answering

Chen Zheng
Michigan State University
zhengc12@msu.edu

Parisa Kordjamshidi
Michigan State University
kordjams@msu.edu

Abstract

This work investigates the challenge of learning and reasoning for Commonsense Question Answering given an external source of knowledge in the form of a knowledge graph (KG). We propose a novel graph neural network architecture, called Dynamic Relevance Graph Network (DRGN). DRGN operates on a given KG subgraph based on the question and answers entities and uses the relevance scores between the nodes to establish new edges dynamically for learning node representations in the graph network. This explicit usage of relevance as graph edges has the following advantages, a) the model can exploit the existing relationships, re-scale the node weights, and influence the way the neighborhood nodes' representations are aggregated in the KG subgraph, b) It potentially recovers the missing edges in KG that are needed for reasoning. Moreover, as a byproduct, our model improves handling the negative questions due to considering the relevance between the question node and the graph entities. Our proposed approach shows competitive performance on two QA benchmarks, CommonsenseQA and OpenbookQA, compared to the state-of-the-art published results.

1 Introduction

Solving Question Answering (QA) problems usually requires both language understanding and human commonsense knowledge. Large-scale pre-trained language models (LMs) have achieved success in many QA benchmarks (Rajpurkar et al., 2016, 2018; Min et al., 2019; Yang et al., 2018). However, LMs have difficulties in predicting the answer when reasoning over external knowledge is required (Yasunaga et al., 2021; Feng et al., 2020).

Therefore, using the external sources of knowledge explicitly in the form of knowledge graphs (KGs) is a recent trend in question answering models (Lin et al., 2019; Feng et al., 2020). Figure 1, taken from the CommonsenseQA benchmark, shows an example for which answering the question requires commonsense reasoning. In this example, the external KG provides the required background information to obtain the reasoning chain from question to answer. We highlight two issues in the previous approaches taken to solve this QA problem: a) the extracted KG subgraph sometimes misses some edges between entities, which breaks the chain of reasoning that the current models can not exploit the connections, b) the semantic context of the question and connection to the answer is not used properly, for example, reasoning when negative terms exist in the question, such as no and not, is problematic.

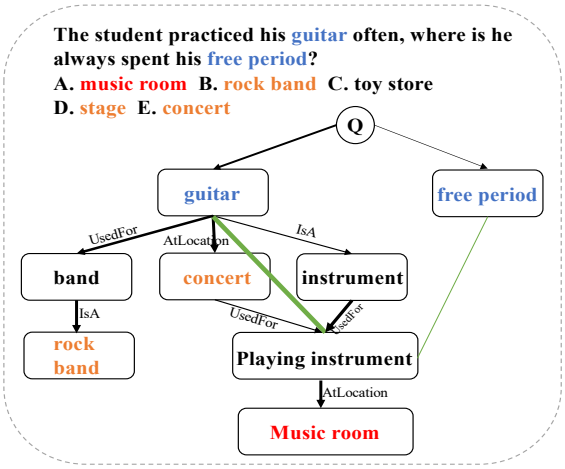

Figure 1: An example of the CommonsenseQA benchmark. Given the question node Q, question entity nodes (blue boxes), correct answer entity node (red box), and wrong answer entity nodes (orange boxes), we predict the answer by reasoning over the question and the extracted KG subgraph.

The first above-mentioned issue is caused by the following reasons. First, the knowledge graph is originally imperfect and does not include the required edges. Second, when constructing the subgraph, to reduce the graph size, most of the models select the entities that appear in two-hop

Proceedings of the 29th International Conference on Computational Linguistics, pages 1357–1366
October 12–17, 2022.

paths (Feng et al., 2020). Therefore, some intermediate concept (entity) nodes and edges are missed in the extracted KG subgraph. In such cases, the subgraph does not contain a complete chain of reasoning. Third, the current models often cannot reason over paths when there is no direct connection between the involved concepts. While finding the chain of reasoning in QA is challenging in general (Zheng and Kordjamshidi, 2021), here this problem is more critical when the KG is the only source and there are missing edges. Looking back at Figure 1, the KG subgraph misses the direct connection between *guitar* and *playing instrument* (green arrow). For the issue of considering question semantics, as (Lin et al., 2019) points out, previous models are not sensitive to the negation words and consequently predict opposite answers. QA-GNN (Yasunaga et al., 2021) model is the first work to deal with the negative questions. QA-GNN improves the reasoning under negation, to some extent, by adding the QA global node to the graph. However, the challenge still exists.

To solve the above challenges, we propose a novel architecture, called Dynamic Relevance Graph Network (DRGN). The motivation of our proposed DRGN is to recover the missing edges and establish direct connections between concepts to facilitate multi-hop reasoning. In particular, DRGN model uses a relational graph network module while influencing the importance of the neighbor nodes using an additional relevance matrix. It potentially can recover the missing edges to establish a direct connection based on the relevancy of the node representations in the KG during the training. The module can potentially capture the connections between distant nodes while benefiting from the existing KG edges. Our proposed model learns representations directly based on the relevance scores between subgraph entity pairs that are computed by Inner Product operation. At each convolutional layer of the graph neural network, we compute the inner product of the nodes based on their current layer's node representations dynamically and build the neighborhoods based on this relevance measure and form a relevance matrix accordingly. This can be seen as a way to learn new edges as the training goes forward in each layer while influencing on the weights of the neighbors dynamically based on their relevance. As shown in Figure 1, the relevance score between *guitar* and *playing instrument* is stronger than other nodes in the subgraph. Moreover, since the graph includes the question node, the relevance between the question node and entity nodes is computed at every layer, making use of the contextual information more effectively. It becomes more clear that the student will spend the *free period* in the *music room* rather than the *concert.*

In summary, the contributions of this work are as follows: **1)** The Proposed DRGN architecture exploits the existing edges in the KG subgraph while explicitly uses the relevance between the nodes to establish direct connections and recover the possibly missing edges dynamically. This technique helps in capturing the reasoning path in the KG for answering the question.
2) Our model exploits the relevance between question and the graph entities, which helps considering the semantics of the question explicitly in the graph reasoning and boosting the performance. In particular, it improves dealing with the negation.
3) Our proposed model obtains competitive results on both CommonsenseQA and OpenbookQA benchmarks. Our analysis demonstrates the significance and effectiveness of the DRGN model.

2 Related Work

2.1 QA using Knowledge Graph

Augmenting QA systems with external knowledge has been studied in many recent research papers. In this direction, pre-trained language models are often employed because they potentially can serve as implicit knowledge bases. (Devlin et al., 2019; Rajaby Faghihi and Kordjamshidi, 2021). To consider more interpretable knowledge, KGs are utilized in the QA models (Zheng and Kordjamshidi, 2022; Feng et al., 2020). However, given that the KGs are usually large and contain many nodes that are irrelevant to the question, the QA models can not effectively use the KG's information (Feng et al., 2020). Moreover, with larger KGs, the computational complexity of learning over them will increase. To deal with this issue, pruning KG nodes based on a variety of metrics has been proposed (Defferrard et al., 2016; Zhou et al., 2020; Velickovic et al., 2018; Hamilton et al., 2017; Ying et al., 2018).

Furthermore, the textual context is used as an additional node in the KG subgraph. For example, (Koncel-Kedziorski et al., 2019) and (Yasunaga et al., 2021) introduce the sentence node into the graph, while (Fang et al., 2020) and (Zheng and Kordjamshidi, 2020) add the paragraph node and

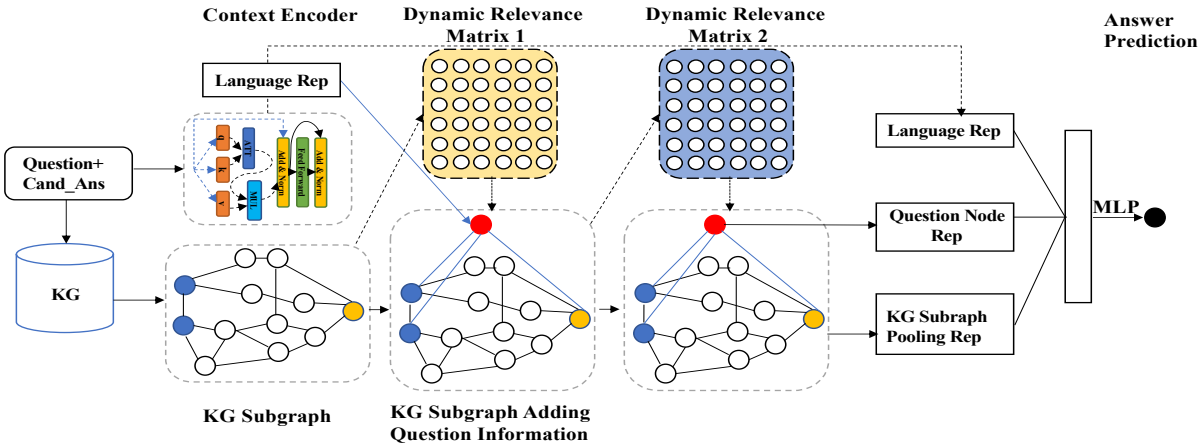

Figure 2: Our proposed DRGN model is composed of the Language Context Encoder module, KG Subgraph Construction module, Graph Neural Network module, and Answer Prediction module. The blue color entity nodes represent the entity mentioned in the question. The yellow color node represents the answer node. The red color node is the question node. We use different colors to draw the dynamic relevance matrix 1 and 2 because the relevance matrix changes dynamically in each graph neural layer.

sentence node to construct a hierarchical graph structure. In this work, we also use the question node as the external node and add it to the KG subgraph. Therefore, the graph representations can learn more contextual information by computing the relevance between the question node and graph entity nodes.

2.2 Graph Neural Networks

Graph convolutional network (GCN) (Kipf and Welling, 2017) is a classic multi-layer graph neural network. The node representations in the graph are strongly related to their neighborhood nodes and edges. For each layer of GCN, the node representations capture the information of their neighborhood nodes and edges via message passing and graph convolutional operation. R-GCN is a variation of GCN that deals with the multi-relational graph structure (Schlichtkrull et al., 2018). (Li et al., 2018) proposes an Adaptive Graph Convolution Network (AGCN) to learn the underlying relations and learn the residual graph Laplacian to improve spectral graph performance. Meanwhile, some varients of GCN try to replace the graph Fourier transform. For example, Graph Wavelet Neural Network (GWNN) (Xu et al., 2019) applies the graph wavelet transform to the graph, and achieves better performance compared to the graph Fourier transform in some tasks.

Meanwhile, several models use the attention based Transformer operator on the graphs. For example, the graph attention network (GAT) (Velickovic et al., 2018) uses the self-attention method and multi-head attention strategy to learn the node representations that consider the neighbors of the nodes. Besides, the gated attention network (GaAN) (Zhang et al., 2018) uses self-attention to aggregate the different heads' information. GaAN utilizes the gate mechanism to replace the average operation that is commonly used in the GAT model.

Dynamic GCN (Ye et al., 2020) is another branch of the GCN family. The dynamic graphs are constructed for different input samples. Moreover, Dynamic GCN learns the dynamic graph structure by a context-encoding network, which takes the whole feature map from the convolution neural network as input and directly predicts the adjacency matrix. Unlike these works, our DRGN model maintains the graph structure statically, but computes the relevance edges dynamically and uses the relevance to weight the neighbors for learning node representations. Besides, our approach uses the existing relationships in the KG, recovers the missing edges and establishes the direct connections by computing the relevance between nodes dynamically. We consider this as learning new edges based on the relevancy of the nodes while the training goes forward in each layer.

3 Dynamic Relevance Graph Network

3.1 Task Formulation

The task of QA over pure knowledge is to choose a correct answer a_{ans} from a set of N candidate answers $\{a_1, a_2, ..., a_n\}$ given input question q and

an external knowledge graph (KG). In fact, the input to the problem is not the whole KG but a subgraph, $G_{sub} = (V, E)$, is selected based on previous research in (Feng et al., 2020) and (Yasunaga et al., 2021). The node set V represents entities in the knowledge subgraph, and the edge set E represents the edges between entities.

3.2 Model Description

Figure 2 shows the proposed Dynamic Relevance Graph Network (DRGN) architecture. Our DRGN includes four modules: Language Context Encoder module, KG Subgraph Construction module, Graph Neural Network module, and Answer Prediction module. In this section, we describe the details of our approach and the way we train our model efficiently.

3.3 Language Context Encoder

For every question q and candidate answer a_i pair, we concatenate them to form Language Context L:

$$L = [[CLS]; q; [SEP]; a_i],$$

where [CLS] and [SEP] are the special tokens used by large-scale pre-trained Language Models (LMs). We feed input L to a pre-trained LMs encoder to obtain the list of token representations $h_L \in \mathbb{R}^{|L|*d}$, where $|L|$ represents the length of the sequence. Then we use the [CLS] representation, denoted as $h_{[CLS]} \in \mathbb{R}^d$, as the representation of L.

3.4 KG Subgraph Construction

We use ConceptNet (Speer et al., 2017), a general-domain knowledge graph, as the commonsense KG. ConceptNet graph has multiple semantic relational edges, e.g., HasProperty, IsA, AtLocation, etc. We follow (Feng et al., 2020) work to construct the subgraphs from KG for each example. The approach is to construct a subgraph from KG that contains the entities mentioned in the question and answer choices. The entities are selected with the exact match between n-gram tokens and ConceptNet concepts using some normalization rules. Then another set of entities is added to the subgraph by following the KG paths of two hops of reasoning based on the current entities in the subgraph.

Furthermore, we add the semantic context of the question as a separate node to the subgraph. This node provides an additional question context to the KG subgraph, G_{sub}, as suggested by (Yasunaga et al., 2021). We link the question node to entity nodes mentioned in the question. The semantic context of the question node Q is initialized by the [CLS] representation described in Section 3.3. The initial representation of the other entities is derived from applying RoBERTa and pooling over their contained tokens (Feng et al., 2020).

3.5 Graph Neural Network Module

The basis of our learning representation is Multi-relational Graph Convolutional Network (R-GCN) (Schlichtkrull et al., 2018). R-GCN is an extension of GCN that operates on a graph with multi-relational edges between nodes. In our case, the relation types between entities are taken from the 17 semantic relations from ConceptNet. Meanwhile, an additional type is added to represent the relationship between the question node and question entities, making the graph structure different from previous works. We denote the set of relations as R.

Our dynamic relevance graph network (DRGN) architecture is the variation of the R-GCN model. To establish the direct connection between the graph nodes and re-scale the importance of the neighbors, we compute the relevance score between the nodes dynamically at each graph layer based on their current learned representations. Then we build the neighborhoods based on this relevance measure and form a relevance matrix, M_{rel}, accordingly. This can be seen as a way to learn new edges based on the relevance of the nodes as the training goes forward in each graph layer. We use inner product to compute the relevance matrix:

$$M_{rel}^{(l)} = h^{(l)\top} h^{(l)} \in \mathbb{R}^{(|V|+1)*(|V|+1)},$$

where $|V|$ is the graph entity nodes sizes, and 1 is added due to using the question node in the graph. The relevance matrix re-scales the weights and influences the way the neighborhood nodes' representations are aggregated in the R-GCN model. M_{rel} is computed dynamically, and the relevance scores change while the representations are computed at each graph layer. In our proposed relational graph, the forward-pass message passing updates of the nodes, denoted by h_i, are calculated as follows:

$$h_i^{(l+1)} = \sigma\Big(\sum_{r \in R} \sum_{j \in \mathbb{N}_i^r} \frac{1}{d_{i,r}} W_r^{(l)} \cdot (M_{rel_{i,j}}^{(l)} h_j^{(l)}) + W_0^{(l)} \cdot (M_{rel_{i,i}}^{(l)} h_i^{(l)})\Big) \in \mathbb{R}^d,$$

where $\mathbb{N}_i^r$ represents the neighbor nodes of node i under relation r, $r \in R$. σ is the activation function,

W_r denotes the learnable parameters. Besides, we calculate the updated question node representation as follows,

$$h_Q^{(l+1)} = \sigma(\sum_{j \in \mathbb{N}^Q} W_Q^{(l)} \cdot F_c([h_Q^{(l)}; (M_{rel_{Q,j}}^{(l)} h_j^{(l)})]) + W_0^{(l)} \cdot (M_{rel_{Q,Q}}^{(l)} h_Q^{(l)})) \in \mathbb{R}^d,$$

where F_c is a two-layer MLP, h_Q is the question node representation. Finally, we stack the node representations to form $h'^{(l+1)}$:

$$h'^{(l+1)} = [h_0^{(l+1)}; h_1^{(l+1)}; \cdots; h_{|V|}^{(l+1)}; h_Q^{(l+1)}] \in \mathbb{R}^{(|V|+1)*d}.$$

We then compute the $(l+1)$ layer's dynamic relevance matrix $M_{rel}^{(l+1)}$ that shows the relevance scores of node representations. Finally, we use the $M_{rel}^{(l+1)}$ to multiply the node representation matrix $h'^{(l+1)}$ that helps the node representation to learn the weights of the edges based on the learned relevance and specifically to include the additional relevance edges between the nodes during the massage passing as follows:

$$h^{(l+1)} = \sigma\left(M_{rel}^{(l+1)} \cdot h'^{(l+1)} \cdot W_g\right) \in \mathbb{R}^{(|V|+1)*d},$$

where W_g is the learnable parameters.

3.6 Answer Prediction

Given the Language Context L and KG subgraph, we use the information from both the language representation $h_{[CLS]}$, question node representation h_Q learnt from the KG subgraph, and the KG subgraph representation pooled from the last graph layer, $pool(h_{G_{sub}})$, to calculate the scores of the candidate answers as follows:

$$p(a|L, G_{sub}) = f_{out}([h_{[CLS]}; h_Q; pool(h_{G_{sub}})]),$$

where f_{out} is a two-layer MLP. Finally, we choose the highest scored answer from N candidate answers as the prediction output. We use the cross entropy loss to optimize the end-to-end model.

4 Experiments and Results

4.1 Datasets

We evaluate our model on two different QA benchmarks, CommonsenseQA and OpenbookQA. Both benchmarks come with an external knowledge graph. We apply ConceptNet to the external knowledge graph on these two benchmarks.

CommonsenseQA (Talmor et al., 2019) is a QA dataset that requires human commonsense reasoning capacity to answer the questions. Each question in CommonsenseQA has five candidate answers without any extra information. The dataset consists of 12,102 questions.

OpenbookQA (Mihaylov et al., 2018) It is a multiple-choice QA dataset that requires reasoning with commonsense knowledge. The OpenbookQA benchmark is a well-defined subset of science QA (Clark et al., 2018) that requires finding the chain of commonsense reasoning to answer a question. Each data sample includes the question, scientific facts, and candidate answers. In our experimental setting, the scientific facts are added to the question part. This makes the problem formulation consistent with the CommonsenseQA setting.

4.2 Implementation Details

We implemented our DRGN architecture using PyTorch.[1] We use the pre-trained RoBERTa-large (Liu et al., 2019) to encode the question. We use cross-entropy loss and RAdam optimizer (Liu et al., 2020) to train our end-to-end architecture. The batch size is set to 16, and the maximum text input sequence length set to 128. Our model uses an early stopping strategy during the training. We use a 3-layer graph neural module in our experiments. Section 5.3 describes the effect of the different number of layers. The learning rate for the LMs is $1e-5$, while the learning rate for the graph module is $1e-3$.

4.3 Baseline Description

KagNET (Lin et al., 2019) is a path-based model that models the multi-hop relations by extracting relational paths from Knowledge Graph and then encoding paths with an LSTM sequence model.

MHGRN (Feng et al., 2020): Multi-hop Graph Relation Network (MHGRN) is a strong baseline. MHGRN model applies LMs to the question and answer context encoder, uses GNN encoder to learn graph representations, and chooses the candidate answer by these two encoders.

QA-GNN (Yasunaga et al., 2021) is the recent SOTA model that uses a working graph to train language and KG subgraph. The model jointly reasons over the question and KG and jointly updates the representations. QA-GNN uses GAT as the

[1]Our code is available at https://github.com/HLR/DRGN.

Models	Dev ACC%	Test ACC%
RoBERTa-no KG	69.6%	67.8%
R-GCN	72.6%	68.4%
GconAttn	72.6%	68.5%
KagNet	73.3%	69.2%
RN	73.6%	69.5%
MHGRN	74.4%	71.1%
QA-GNN	76.5%	73.4%
DRGN	**78.2%**	**74.0%**

Table 1: Dev accuracy and Test accuracy (In-House split) of various models on the CommonsenseQA benchmark, following by (Lin et al., 2019).

Models	Dev	Test
RoBERTa-large	66.7%	64.8%
R-GCN	65.0%	62.4%
GconAttn	64.5%	61.9%
RN	66.8%	65.2%
MHGRN	68.1 %	66.8%
QA-GNN	68.9 %	67.8%
DRGN	**70.1%**	**69.6%**
AristoRoBERTaV7	79.2%	77.8%
T5(3 Billion Parameters)	-	83.2%
UnifiedQA(11 Billion Parameters)	-	**87.2%**
AristoRoBERTaV7+MHGRN	78.6%	80.6%
AristoRoBERTaV7+QA-GNN	80.4%	82.8%
AristoRoBERTaV7+DRGN	**81.8%**	**84.1%**

Table 2: Development and Test accuracy of various model performance on the OpenbookQA benchmark.

backbone to do message passing on the graph. To learn the semantic edge information, QA-GNN directly adds the edge representation to the local node representation and cannot learn the global structure of the edges, which is inefficient. However, our model uses the global multi-relational adjacency matrices to learn the edge information.

4.4 Result Comparison

Table 1 shows the performance of different models on the CommonsenseQA benchmark. KagNet and MHGRN are two strong baselines. Our model outperforms the KagNet by 4.8% and MHGRN by 2.9% on CommonsenseQA benchmark. This result shows the effectiveness of our DRGN architecture. Table 2 shows the performance on the OpenbookQA benchmark. There are a few recent papers that exploit larger LMs, such as T5 (Raffel et al., 2020) that contains 3 billion parameters (10x larger than our model,) and UnifiedQA (Khashabi et al., 2020) (32x larger). For a fair comparison, we use the same RoBERTa setting for the input representation when we evaluate OpenbookQA. Our model performance, potentially, will be improved after using these larger LMs. To demonstrate this point, we did additional experiments using AristoRoBERTaV7 (Clark et al., 2019) as a backbone to train our model. Our model achieves better performance when using the larger LMs compared to other baseline models. The performance shows that the more implicit information learned from pre-trained language models, the more effective relevance information established between graph nodes. We should note that GREASELM (Zhang et al., 2022) and GSC (Wang et al., 2022) are two most recent models that are developed in parallel with our DRGN. GREASELM aims to ground language context in commonsense knowledge graph by fusing token representations from pretrained LMs and GNN over *Modality Interaction* layers (Zhang et al., 2022). GSC designs a *Graph Soft Counter* layer (Wang et al., 2022) to enhance the graph reasoning capacity. Our results are competitive with the reported ones in those parallel works while each work emphasizes different contributions.

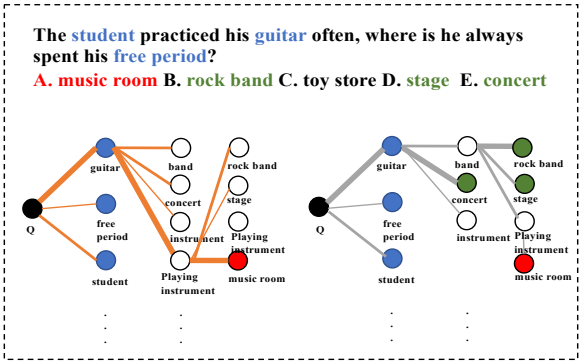

Figure 3: The complete reasoning chain from the question node to the candidate answer node. The blue nodes are question entity nodes, the red and green nodes are the candidate answer nodes. The thicker edges indicate the higher relevance score to the neighborhood node, while the thinner edges indicate the lower score. The left side is the reasoning chain selected from our model (orange edges), while the right side is selected from the baseline models (grey edges).

5 Analysis

5.1 Effects on Finding the Line of Reasoning

In this section, we analyze the effectiveness of our DRGN model that helps in recovering the missing edges and establishing direct connections based on the relevancy of the node representations in the KG.

Models	Test ACC % (Overall)	Test ACC% question w/ negative
RoBERTa-large	68.7 %	54.2%
KagNet	69.2 %	54.2 %
MHGRN	71.1 %	54.8%
QA-GNN	73.4 %	58.8%
DRGN	**75.0%**	**60.1%**

Table 3: Performance on questions with negation in In-house split test CommonsenseQA.

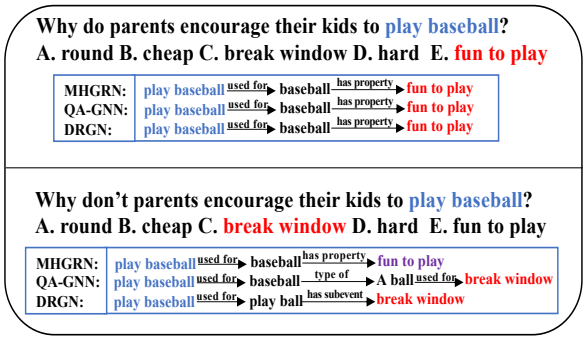

Figure 4: The case study of the negation examples. The question in the bottom box includes the negation words. The red colored text represents the gold answer, and the purple colored represents the wrong answer. In the blue box, each line represents the commonsense reasoning chain of each model.

As we described in Section 3.4, to keep the graph size small, most of the models construct the KG subgraph via selecting the entities that appear in two-hop paths. Therefore, some intermediate concept nodes and edges are missed in the extracted KG subgraph, and the complete reasoning chain from the question entity node to the candidate answer node can not be found.

For example, as shown in Figure 3, the question is "The student practiced his guitar often, where is he always spent his free period?" and the answer is "music room". The reasoning chain includes 2 hops, that is, "guitar → playing instrument → music room". Since the constructed graph misses the direct edge between "guitar" and "playing instrument", MHGRN and QA-GNN baselines select the wrong intermediate node and predict the wrong answer "concert" and "rock band" by the grey edges described in the Figure 3. In contrast, our DRGN model makes a correct prediction by computing the relevance score of the nodes based on their learned representations and forming new edges accordingly. As we describe in Section 3.4, our model initializes the entity node representation by large-scale pre-trained language models (LMs). The implicit representations of LMs are learned from the huge corpora, and the knowledge is implicitly learned. Therefore, these two entities, "guitar" and "playing instrument", start with an implicit connection. By looking at the relevance changes, after several layers of graph encoding, the relevance score between "guitar" and "playing instrument" becomes stronger. In contrast, the relevance score between "guitar" and "concert" becomes weaker because of the contextual information "free period". This is the primary reason why our DRGN model obtains the correct reasoning chain.

5.2 Effects on Semantic Context

While the graph has a broad coverage of knowledge, the semantic context of the question and connection to the answer is not used properly. For example, dealing with negation can not perform well (Yasunaga et al., 2021). Since our dynamic relevance matrix includes the semantic context of the question, the relevance between the question and graph entities is computed at every graph neural layer while considering the negation in the node representations. Intuitively, this should improve handling the negative question in our model.

To analyze this hypothesis for DRGN architecture, we compare the performance of various models on questions containing negative words (e.g., no, not, nothing, unlikely) from CommonsenseQA following (Yasunaga et al., 2021). The result is shown in Table 3. We observe that the baseline models of KagNet and MHGRN provide limited improvements over RoBERTa on questions with negation words (+0.4%). However, our DRGN model exhibits a huge boost (+5.9%). Moreover, the DRGN model gains a larger improvement in the accuracy compared to the QA-GNN model, demonstrating the effectiveness of considering relevance between question semantics and graph entity that experimentally confirms our hypothesis. An additional ablation study in Table 5 confirms this idea further. When removing the question information from DRGN, we observe that the performance on negation becomes close to the MHGRN.

Figure 4 shows qualitative examples about the positive and negative questions. For the positive question, all the models obtain the same reasoning chain "play baseball-(used for)→ baseball-(has property)→ fun to play", including MHGRN, QA-GNN, and our architecture. However, when adding the negative words, MHGRN obtains the same rea-

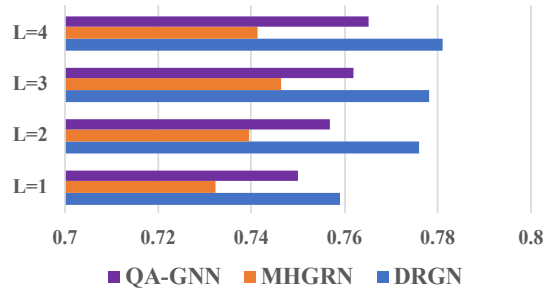

Figure 5: The Effect of number of layers in QA-GNN, MHGRN, and DRGN models on CommonsenseQA.

Models	Time	Space
l-layer KagNet	$O(\|R\|^l\|V\|^{l+1}l)$	$O(\|R\|^l\|V\|^{l+1}l)$
l-layer MHGRN	$O(\|R\|^2\|V\|^2l)$	$O(\|R\|\|V\|l)$
l-layer QA-GNN	$O(\|V\|^2l)$	$O(\|R\|\|V\|l)$
l-layer DRGN	$O(\|R\|^2\|V\|^2l)$	$O(\|R\|\|V\|^2l)$

Table 4: The time complexity and space complexity comparison between DRGN and baseline models.

soning chain as the positive situation, while QA-GNN and DRGN find the correct reasoning chain. One interesting finding is that DRGN can detect the direct connection using fewer hops to establish the reasoning chain.

5.3 Effects of Number of Graph Layers

The number of graph layers is an influencing factor for DRGN architecture because our relevance matrix is computed dynamically, and the relevance scores change while the representations are computed at each graph layer. We evaluate the effects of multiple layers l for the baseline models and our DRGN by evaluating its performance on the CommonsenseQA. As shown in Figure 5, the increase of l continues to bring benefits until $l = 4$ for DRGN. We compare the performance after adding each layer for MHGRN, QA-GNN, and our DRGN. We observe that DRGN consistently achieves the best performance with the same number of layers as the baselines.

Table 4 shows the time complexity and the space complexity comparison between DRGN model and baseline model. We compare the computational complexity based on the number of layers l, the number of nodes V, and the number of relations R. Our model and MHGRN have the same time complexity because both models use the R-GCN model as the backbone. Besides, QA-GNN directly adds the edge representation to the local node representation during the graph pre-processing step and learns the graph node representation without

Models	Dev ACC
DRGN w/o KG subgraph	69.6%
+ KG subgraph	72.6%
+ relational edges in graph	73.7%
+ question node in graph	74.9%
+ dynamic relevance matrix	78.2%

Table 5: Ablation Study on CommonsenseQA dataset.

the global semantic relational adjacency matrices. After adding the dynamic relevance matrix at each graph layer, our DRGN model achieves better performance compared to other baseline architectures. For the space complexity, our model's space complexity is slightly larger than MHGRN because DRGN introduces the extra dynamic relevance matrix. However, this cost depends on the size of the subgraph, which is usually small while it leads to a huge improvement.

5.4 Ablation of DRGN Modules

To evaluate the effectiveness of various components of DRGN, we perform an ablation study on the CommonsenseQA development benchmark. Table 5 shows the results of ablation study. First, we remove the whole commonsense subgraph. Our model without the subgraph obtains 69.6% on the CommonsenseQA. This shows how the implicit language model can answer the questions without the external KG, which is not high-performing but yet impressive. After adding the KG subgraph, the accuracy improves to 72.6% on the CommonsenseQA benchmark. Second, we keep the KG subgraph and add multiple relational edge information from the subgraph (described in section 3.5). Without the relational edges, the accuracy becomes 73.7%. This result shows that the multiple relational edges help in learning better graph node representations and obtaining a higher performance. Third, we keep the multi-relational subgraph and add the question node. In other words, we incorporate the semantic relationship between the question node and the graph entities. The accuracy of the model improves to 74.9%. Finally, we add the most important component, the dynamic relevance matrix, to each graph layer. The large improvement demonstrates the importance of the dynamic relevance matrix and the effectiveness of DRGN architecture.

6 Conclusion

In this paper, we propose a novel Dynamic Relevance Graph Network (DRGN) architecture for commonsense question answering given an external source of knowledge in the form of a Knowledge Graph. Our model learns the graph node representation while a) exploits the existing relations in KG, b) re-scales the importance of the neighbor nodes in the graph based on training a dynamic relevance matrix, c) establishes direct connections between graph nodes based on measuring the relevance scores of the nodes dynamically during training. The dynamic relevance edges help in finding the chain of reasoning when there are missing edges in the original KG. Our quantitative and qualitative analysis shows that the proposed approach facilitates answering the complex questions that need multiple hops of reasoning. Furthermore, since DRGN uses the relevance between the question node and graph entities, it exploits the richer semantic context of the question in graph reasoning which leads to improvements in the performance on the negative questions. Our proposed approach shows competitive performance on two QA benchmarks, including CommonsenseQA and OpenbookQA.

Acknowledgments

We thank all reviewers for their suggestions and helpful comments. This project is supported by National Science Foundation (NSF) CAREER award #2028626 and partially supported by the Office of Naval Research (ONR) grant #N00014-20-1-2005. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation nor the Office of Naval Research.

References

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Peter Clark, Oren Etzioni, Daniel Khashabi, Tushar Khot, Bhavana Dalvi Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, Niket Tandon, et al. 2019. From'f'to'a'on the ny regents science exams: An overview of the aristo project. *arXiv preprint arXiv:1909.01958*.

M. Defferrard, X. Bresson, and P. Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yuwei Fang, S. Sun, Zhe Gan, Rohit Radhakrishna Pillai, Shuohang Wang, and Jingjing Liu. 2020. Hierarchical graph network for multi-hop question answering. In *EMNLP*.

Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable multi-hop relational reasoning for knowledge-aware question answering. In *EMNLP*.

William L. Hamilton, Zhitao Ying, and J. Leskovec. 2017. Inductive representation learning on large graphs. In *NIPS*.

D. Khashabi, S. Min, T. Khot, A. Sabhwaral, O. Tafjord, P. Clark, and H. Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. *EMNLP - findings*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. In *NAACL*.

Ruoyu Li, S. Wang, Feiyun Zhu, and J. Huang. 2018. Adaptive graph convolutional neural networks. In *AAAI*.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *NAACL*.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2020. On the variance of the adaptive learning rate and beyond. In *ICLR*.

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*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on*

Empirical Methods in Natural Language Processing, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Multi-hop reading comprehension through question decomposition and rescoring. In *ACL*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Hossein Rajaby Faghihi and Parisa Kordjamshidi. 2021. Time-stamped language model: Teaching language models to understand the flow of events. In *The 2021 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-2021)*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *ACL*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

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 the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR*.

Kuan Wang, Yuyu Zhang, Diyi Yang, Le Song, and Tao Qin. 2022. Gnn is a counter? revisiting gnn for question answering. In *ICLR*.

Bingbing Xu, Huawei Shen, Qi Cao, Yunqi Qiu, and Xueqi Cheng. 2019. Graph wavelet neural network. In *ICLR*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. In *NAACL*.

Fanfan Ye, Shiliang Pu, Qiaoyong Zhong, Chao Li, Di Xie, and Huiming Tang. 2020. Dynamic gcn: Context-enriched topology learning for skeleton-based action recognition. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 55–63.

Rex Ying, Ruining He, K. Chen, Pong Eksombatchai, William L. Hamilton, and J. Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and D. Yeung. 2018. Gaan: Gated attention networks for learning on large and spatiotemporal graphs. In *UAI*.

Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D. Manning, and Jure Leskovec. 2022. Greaselm: Graph reasoning enhanced language models. In *ICLR*.

Chen Zheng and Parisa Kordjamshidi. 2020. SRLGRN: Semantic role labeling graph reasoning network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8881–8891, Online. Association for Computational Linguistics.

Chen Zheng and Parisa Kordjamshidi. 2021. Relational gating for "what if" reasoning. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4015–4022. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Chen Zheng and Parisa Kordjamshidi. 2022. Relevant CommonSense subgraphs for "what if..." procedural reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1927–1933, Dublin, Ireland. Association for Computational Linguistics.

Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and M. Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81.