SRLGRN: Semantic Role Labeling Graph Reasoning Network

Chen Zheng
Michigan State University
zhengc12@msu.edu

Parisa Kordjamshidi
Michigan State University
kordjams@msu.edu

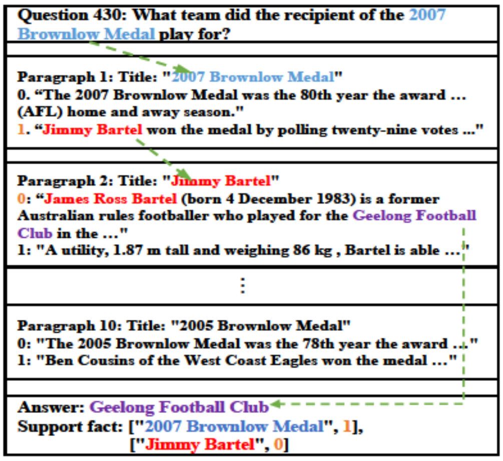
Abstract

This work deals with the challenge of learning and reasoning over multi-hop question answering (QA). We propose a graph reasoning network based on the semantic structure of the sentences to learn cross paragraph reasoning paths and find the supporting facts and the answer jointly. The proposed graph is a heterogeneous document-level graph that contains nodes of type sentence (question, title, and other sentences), and semantic role labeling sub-graphs per sentence that contain arguments as nodes and predicates as edges. Incorporating the argument types, the argument phrases, and the semantics of the edges originated from SRL predicates into the graph encoder helps in finding and also the explainability of the reasoning paths. Our proposed approach shows competitive performance on the HotpotQA distractor setting benchmark compared to the recent state-of-the-art models.

1 Introduction

Understanding and reasoning over natural language plays a significant role in artificial intelligence tasks such as Machine Reading Comprehension (MRC) and Question Answering (QA). Several QA tasks have been proposed in recent years to evaluate the language understanding capabilities of machines (Rajpurkar et al., 2016; Joshi et al., 2017; Dunn et al., 2017). These tasks are single-hop QA tasks and consider answering a question given only one single paragraph. Many existing neural models rely on learning context and type-matching heuristics (Weissenborn et al., 2017). Those rarely build reasoning modules but achieve promising performance on single-hop QA tasks. The main reason is that these single-hop QA tasks are lacking a realistic evaluation of reasoning capabilities because they do not require complex reasoning.

Recently multi-hop QA tasks, such as HotpotQA (Yang et al., 2018) and WikiHop (Welbl et al., 2018), have been proposed to assess multi-hop reasoning ability. HotpotQA task provides annotations to evaluate document level question answering and finding supporting facts. Providing supervision for supporting facts improves explainabilty of the predicted answer because they clarify the cross paragraph reasoning path. Due to the requirement of multi-hop reasoning over multiple documents with strong distraction, multi-hop QA tasks are challenging. Figure 1 shows an example of HotpotQA. Given a question and 10 paragraphs, only paragraph 1 and paragraph 2 are relevant. The second sentence in paragraph 1 and the first sentence in paragraph 2 are the supporting facts. The answer is “Geelong Football Club”.


Question 430: What team did the recipient of the 2007 Brownlow Medal play for?

Paragraph 1: Title: "2007 Brownlow Medal"
0. “The 2007 Brownlow Medal was the 80th year the award ... (AFL) home and away season."
1. “Jimmy Bartel won the medal by polling twenty-nine votes ..."

Paragraph 2: Title: "Jimmy Bartel"
0: “James Ross Bartel (born 4 December 1983) is a former Australian rules footballer who played for the Geelong Football Club in the ..."
1: "A utility, 1.87 m tall and weighing 86 kg , Bartel is able ..."

⋮

Paragraph 10: Title: "2005 Brownlow Medal"
0: "The 2005 Brownlow Medal was the 78th year the award ..."
1: "Ben Cousins of the West Coast Eagles won the medal ..."

Answer: Geelong Football Club
Support fact: ["2007 Brownlow Medal", 1],
["Jimmy Bartel", 0]

Figure 1: An example of HotpotQA data.

Primary studies in HotpotQA task prefer to use a reading comprehension neural model (Min et al., 2019; Zhong et al., 2019; Yang et al., 2018). First, they use a neural retriever model to find the relevant paragraphs to the question. After that, a neural reader model is applied to the selected paragraphs for answer prediction. Although these approaches obtain promising results, the performance of evaluating multi-hop reasoning capability is unsatisfac-

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 8881–8891,
November 16–20, 2020. ©2020 Association for Computational Linguistics

tory (Min et al., 2019).

To solve the multi-hop reasoning problem, some models tried to construct an entity graph using Spacy[1] or Stanford CoreNLP (Manning et al., 2014) and then applied a graph model to infer the entity path from question to the answer (Chen et al., 2019; Xiao et al., 2019; Clark and Gardner, 2018; Fang et al., 2019). However, these models ignore the importance of the semantic structure of the sentences and the edge information and entity types in the entity graph. To take the in-depth semantic roles and semantic edges between words into account here we use semantic role labeling (SRL) graph as the backbone of a graph convolutional network. Semantic role labeling provides the semantic structure of the sentence in terms of argument-predicate relationships (He et al., 2018). The argument-predicate relationship graph can significantly improve the multi-hop reasoning results. Our experiments show that SRL is effective in finding the cross paragraph reasoning path and answering the question.

Our proposed semantic role labeling graph reasoning network (SRLGRN) jointly learns to find cross paragraph reasoning paths and answers questions on multi-hop QA. In SRLGRN model, firstly, we train a paragraph selection module to retrieve gold documents and minimize distractor. Second, we build a heterogeneous document-level graph that contains sentences as nodes (question, title and sentence), and SRL sub-graphs including semantic role labeling arguments as nodes and predicates as edges. Third, we train a graph encoder to obtain the graph node representations that incorporate the argument types and the semantics of the predicate edges in the learned representations. Finally, we jointly train a multi-hop supporting fact prediction module that finds the cross paragraph reasoning path, and answer prediction module that obtains the final answer. Notice that both supporting fact prediction and answer prediction are based on contextual semantics graph representations as well as token-level BERT pre-trained representations. The contributions of this work are as follows:

1) We propose the SRLGRN framework that considers the semantic structure of the sentences in building a reasoning graph network. Not only the semantics roles of nodes but also the semantics of edges are exploited in the model.

2) We evaluate and analyse the reasoning capabilities of the semantic role labeling graph compared to usual entity graphs. The fine-grained semantics of SRL graph help in both finding the answer and the explainability of the reasoning path.

3) Our proposed model obtains competitive results on both HotpotQA (Distractor setting) and the SQuAD benchmarks.

2 Related Work

2.1 Graph Models for Multi-Hop Reasoning

Previous QA datasets, such as TriviaQA (Joshi et al., 2017) and SearchQA (Dunn et al., 2017), and MRC datasets, like SQuAD (Rajpurkar et al., 2016), rarely require sophisticated reasoning (such as cross paragraph reasoning) to answer the question and fail to provide ground-truth explanations for answers. Recently, WikiHop (Welbl et al., 2018) and HotpotQA (Yang et al., 2018) are two published multi-hop QA datasets that provide multiple paragraphs. Those QA datasets require a multi-hop reasoning model to learn the cross paragraph reasoning paths and predict the correct answer.

Most of the existing multi-hop QA models (Tu et al., 2019; Xiao et al., 2019; Fang et al., 2019) utilize graph based neural networks, such as graph attention network (Velickovic et al., 2018), graph recurrent network (Song et al., 2018b), and graph convolutional network (Kipf and Welling, 2017). Moreover, multi-hop QA models use different ways to construct entity graphs. Coref-GRN (Dhingra et al., 2018) utilize co-reference resolution to build the entity graph. MHQA-GRN (Song et al., 2018a) is an updated version of Coref-GRN that adds sliding windows. Entity-GCN (Cao et al., 2019) builds the graph using entities and different types of edges called match edges and complement edges. DFGN (Xiao et al., 2019) and SAE (Tu et al., 2019) construct entity graph through named entity recognition (NER).

In contrast to the above mentioned models, our SRLGRN builds a heterogeneous graph that contains a document-level graph of various sentences and replaces the entity-based graphs with argument-predicate based sub-graphs using SRL.

2.2 Semantic Role Labeling

The goal of semantic role labeling is to capture argument and predicate relationships given a sentence, such as “who did what to whom.” Several deep SRL models achieve highly accurate results in

[1] https://spacy.io

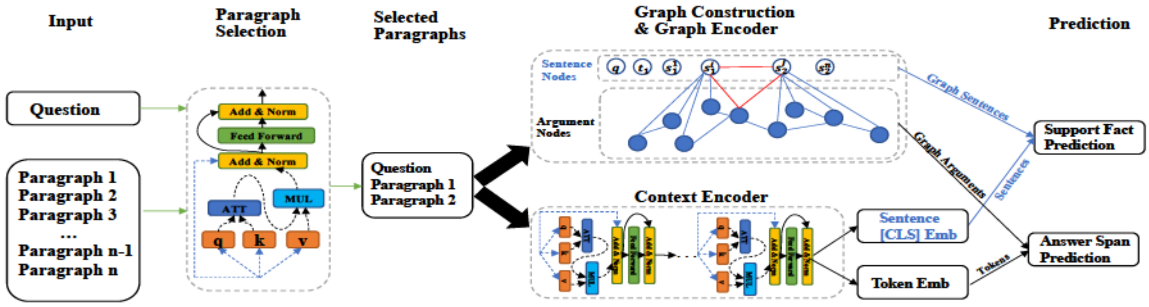


Figure 2: Our proposed SRLGRN model is composed of Paragraph Selection, Graph Construction, Graph Encoder, Supporting Fact prediction, and Answer Span prediction.

finding argument spans (Zhou and Xu, 2015; Tan et al., 2018; Marcheggiani et al., 2017; He et al., 2017). However, those models are evaluated based on given gold predicates. Therefore, some deep models (He et al., 2018; Guan et al., 2019) are proposed to recognize all argument-predicate pairs. Recently, Shi and Lin proposed a BERT Model for SRL and Relation Extraction.

3 Model Description

Our proposed SRLGRN approach is composed of Paragraph Selection, Graph Construction, Graph encoder, Supporting Fact prediction, and Answer Span prediction modules. Figure 2 shows the proposed architecture. In this section, we introduce our approach in detail and then explain how to train it with an efficient algorithm.

3.1 Problem Formulation

Formally, the problem is to predict supporting fact y_{SF} and answer span y_{ans} given input question q and candidate paragraphs. Each paragraph content $\mathcal{C} = \{t, s_1, \ldots, s_n\}$ includes title t and several sentences $\{s_1, \ldots, s_n\}$.

3.2 Paragraph Selection

Most of the paragraphs are distractors in the HotpotQA task (Yang et al., 2018). SRLGRN can select gold documents and minimize distractors from given N documents by a Paragraph Selection module. The Paragraph Selection is based on the pre-trained BERT model (Devlin et al., 2018). Our Paragraph Selection module has two rounds explained in section 3.2.1 and section 3.2.2.

3.2.1 First Round Paragraph Selection

For every candidate paragraph, we take the question q and the paragraph content $\mathcal{C}$ as input:

$$Q_1 = [[CLS]; q; [SEP]; \mathcal{C}], \qquad (1)$$

where Q_1 represents the input, [CLS] and [SEP] are the same as BERT tokenizer process (Devlin et al., 2018). We feed input Q_1 to a pre-trained BERT encoder to obtain token representations. Then we use $BERT_{[CLS]}$ token representation as the summary representation of the paragraph. Meanwhile, we utilize a two-layer MLP to output the relevance score, y_{sel}. The paragraph which obtains the highest relevance score is selected as the first relevant context. We concatenate q to the selected paragraph as q_{new} for the next round of paragraph selection.

3.2.2 Second Round Paragraph Selection

For the remaining $N-1$ candidate paragraphs, we use the same model as first round paragraph selection to generate a relevance score that takes q_{new} and paragraph content as input. We call this process as second round paragraph selection. Similar to section 3.2.1, one of the remaining candidate paragraphs with the highest score is selected. Afterwards, we concatenate the question and the two selected paragraphs to form a new context used as the input text for graph construction.

3.3 Heterogeneous SRL Graph Construction

We build a heterogeneous graph that contains document-level sub-graph $\mathcal{S}$ and argument-predicate SRL sub-graph Arg for each data instance. In the graph construction process, the document level sub-graph $\mathcal{S}$ includes question q, title t_1 and sentences $s_1^{1,\ldots,n}$ from first round selected paragraph, and title t_2 and sentences $s_2^{1,\ldots,n}$ from the second round selected paragraph, that is $\{q, t_1, s_1^1, \ldots, s_1^n, t_2, s_2^1, \ldots, s_2^n\} \in \mathcal{S}$. The argument-predicate SRL sub-graphs Arg, including arguments as nodes and the predicates as edges, are generated using AllenNLP-SRL model (Shi and Lin, 2019). Each argument node is the concatenation of argument phrase and argument type,

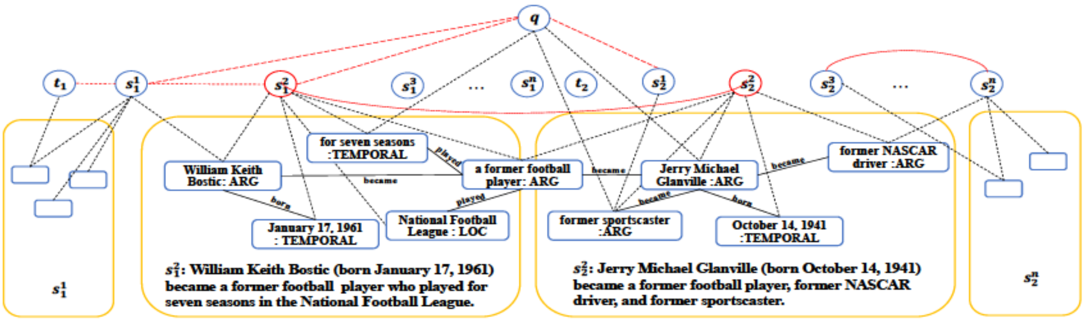

q
t_1
s_1^1
s_1^2
s_1^3
...
s_1^n
t_2
s_2^1
s_2^2
s_2^3
...
s_2^n
for seven seasons :TEMPORAL
played
William Keith Bostic: ARG
became
a former football player: ARG
born
played
January 17, 1961 : TEMPORAL
National Football League : LOC
s_1^1
s_1^2: William Keith Bostic (born January 17, 1961) became a former football player who played for seven seasons in the National Football League.
became
Jerry Michael Glanville :ARG
became
former NASCAR driver :ARG
became
born
former sportscaster :ARG
October 14, 1941 :TEMPORAL
s_2^2: Jerry Michael Glanville (born October 14, 1941) became a former football player, former NASCAR driver, and former sportscaster.
s_2^n

Figure 3: An example of Heterogeneous SRL Graph. The question is "Who is younger Keith Bostic or Jerry Glanville?" The circles show the document-level nodes, i.e., sentences. The blue squares show the argument nodes. The argument nodes include argument phrase and argument type information. The solid black lines are semantic edges between two arguments carrying the predicate information. The black dashed lines show the edges between sentence nodes and argument nodes. The red dashed lines show the edges between two sentences if there exists a shared argument (based on exact string match). The orange blocks are the SRL argument-predicate sub-graphs for sentences. s_i^j means the j-th sentence from the i-th paragraph.

including "TEMPORAL", "LOC", etc.

Figure 3 describes the construction of the heterogeneous graph. The heterogeneous graph's edges are added as follows: **1)** There will be an edge between a sentence and an argument if an argument appears in this sentence (the black dashed lines in Figure 3); **2)** Two sentences s_i and s_j will have an edge if they share an argument by exact matching (the red dashed lines); **3)** Two argument nodes Arg_i and Arg_j will have an edge if a predicate exists between Arg_i and Arg_j (the black solid lines); **4)** There will be an edge between the question and sentence if they share an argument (the red dashed lines).

Figure 3 shows an example of a heterogeneous SRL graph. s_1^2 and s_2^2 are connected because of a shared argument node "a former football player: ARG". Besides, the shared argument node has several semantic edges, such as "played" and "became". In this way, the shared argument node and other connected argument nodes have argument-predicate relationships.

We create two matrices based on the constructed graph that we will use in section 3.4. We build a predicate-based semantic edge matrix K and a heterogeneous edge weight matrix A. The semantic edge matrix K is a matrix that stores the word index of the predicates. We initialize all the elements of K with empty, $\emptyset$. If two argument nodes Arg_i and Arg_j related to the same predicate, we add that predicate word index to $K_{(Arg_i, Arg_j)}$. Sometimes, Arg_i and Arg_j are related to more than one predicate.

In the meantime, the heterogeneous edge weight matrix A is a matrix that stores different types of edge weights. We divide the edges into three types: sentence-argument edges, argument-argument edges, and sentence-sentence edges.

The weight of a sentence-sentence edge is 1 when two sentences share an argument. Meanwhile, the weight of a sentence-argument edge is 1 if there exists an edge between a sentence and an argument. If two argument nodes have an edge, the weight can be calculated by point-wise mutual information (PMI) (Bouma, 2009). The reason we use PMI is that it can better explain associations between nodes compared to the traditional co-occurrence count method (Yao et al., 2019).

3.4 Graph Encoder

Section 3.3 introduces the detailed process of building a heterogeneous graph. Next, we introduce the Graph Convolution Network (Kipf and Welling, 2017) to obtain the graph embeddings. Graph Convolution Network (GCN) is a multi-layer network that uses the graph input directly and generates embedding vectors of the graph.

Besides, GCN plays an essential role in incorporating higher-order neighborhood nodes and helps in capturing the structural graph information. The SRL graph uses the semantic structure of the sentence to form the graph nodes and semantic edges, making the GCN's representation more explainable. For instance, the GCN node vectors of document level sub-graph help in finding the supporting fact path, while GCN node vectors of argument-predicate level sub-graph help in identifying the text span of the potential answers. In this work, we consider a two-layer GCN to allow message passing operations and learn the graph embeddings. The graph embeddings are computed as follows:

$$E_1 = (D^{-\frac{1}{2}} A D^{-\frac{1}{2}})[X_{Arg}; X_{\mathcal{S}}]W_1, \quad (2)$$

$$G = (D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) f(E_1) W_2, \quad (3)$$

where E_1 and G are graph embedding outputs of two GCN layers that incorporate higher-order neighborhood nodes by stacking GCN layers. $f(x)$ is an activation function, D is the degree matrix of the graph (Kipf and Welling, 2017), A is the heterogeneous edge weight matrix, and W_1 and W_2 are the learned parameters. X represents node embeddings, including argument-predicate embedding X_{Arg} and sentence embedding $X_{\mathcal{S}}$. Notice that each argument embedding X^i_{Arg} is the concatenation of the argument node Arg^i embedding and the average embedding of K^i_{Arg}. Given G, we use $G_{\mathcal{S}}$ to represent document level node embeddings, and G_{Arg} to represent argument-predicate level node embeddings.

3.5 Supporting Fact Prediction

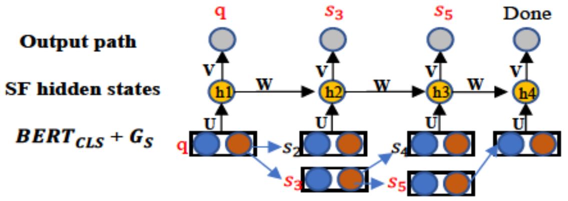


Figure 4: An example of Supporting Fact Prediction.

The goal of supporting fact (SF) prediction is to find the SF that is necessary to arrive at the answer. Inspired by Asai et al., we utilize RNN with a beam search to find the best document-level SF path. This approach turns out to be effective for selecting the SF reasoning path. Notice that, our supporting fact prediction is not only based on BERT and RNN, but also incorporates document level graph node embeddings $G_{\mathcal{S}}$.

Formally, we use the concatenation of the graph sentence embedding, $G_{\mathcal{S}}$ (blue circles in Figure 4), and BERT's [CLS] token representation (orange circles) to represent the candidate sentence $X^{cand}_{\mathcal{S}}$:

$$X^{cand}_{\mathcal{S}} = [G^{cand}_{\mathcal{S}}; BERT_{[CLS]}(q, \mathcal{S}_{cand})], \quad (4)$$

where $\mathcal{S}_{cand}$ represents the neighbors of the candidate sentence. Afterwards, two feed-forward fully connected layers with activation functions determine whether s_{cand} is an actual SF. The process of selecting an SF is shown as follows:

$$h_t = \sigma(W h_{t-1} + U X^{cand}_{\mathcal{S}} + b_h), \quad (5)$$

$$o_t = V h_t + b_o, \quad (6)$$

where h_t is the hidden state of the RNN at the t-th SF reasoning step, σ is the activation function. W, U, V, b_h and b_o are the parameters.

Finally, we use the beam search to output SF paths, choosing the highest scored path as our final supporting fact answer y_{SF}:

$$y_{\text{SF}} = \arg\max_{1 \le t \le T} \prod o_t, \quad (7)$$

where T is the maximum number of reasoning hops. We penalize with the cross-entropy loss. More details are described in section 3.7.

Figure 4 shows an example of the predicted SF process. Based on the constructed heterogeneous graph, two sentence nodes have an edge if they share an argument. We start from question node q as the first input sentence. Since q is a unique input, we select q as the first SF candidate. In the second step, two candidate sentence nodes, s_2 and s_3 that are neighbor nodes of q, are chosen as the input. We separately feed s_2 and s_3 to the RNN layers. The sentence s_3 that obtains a larger logit score is selected as the second SF candidate of the reasoning path. In the third step, s_4 and s_5 are neighbor nodes of the second SF, s_3. Then the model chooses s_5 as the third SF. In the end, s_1, s_3, and s_5 are the supporting facts.

3.6 Answer Span Prediction

The goal of the answer span prediction module is to output "yes", "no", or answer span for the final answer. We firstly design an answer type classification based on BERT and an additional two fully connected feed-forward layers. If the highest probability of type classification is "yes" or "no", we directly output the answer. The input of type classification is $BERT_{[CLS]}$. The answer type y_{type} can be calculated as

$$y_{\text{type}} = MLP_{\text{type}}([BERT_{[CLS]}]).$$

If the answer is not "yes" or "no", we compute the logit of every token to find the start position i and end position j for answer span. The logit is calculated using BERT as the input given to two fully connected layers. The input token representation is the concatenation of BERT token representation $BERT_{tok}$ and graph embedding G_{Arg}. The answer span y_{ans} can be computed as

$$y_{\text{ans}} = \underset{i,j,\, i \le j}{\arg\max}\, y^{i}_{start} y^{j}_{end}, \tag{8}$$

$$y^{i}_{\text{start}} = MLP_{\text{start}}([BERT^{i}_{tok}; G^{i}_{Arg}]), \tag{9}$$

$$y^{i}_{\text{end}} = MLP_{\text{end}}([BERT^{i}_{tok}; G^{i}_{Arg}]), \tag{10}$$

where y_{ans} is the index pair of (start position, end position), y^{i}_{start} represents the logit score of the i-th word as the start position, and y^{i}_{end} represents the logit score of the i-th word as the end position.

3.7 Objective Function

Inspired by Xiao et al. and Tu et al., the joint objective function includes the sum of cross-entropy losses for the span prediction L_{ans}, answer type classification L_{type}, and supporting fact prediction L_{SF}. The loss function is computed as follows:

$$\begin{aligned} L_{\text{joint}} &= L_{\text{ans}} + L_{\text{SF}} + L_{\text{type}} \\ &= \lambda_1(-y_{\text{start}} \log y_{\text{start}} - y_{\text{end}} \log y_{\text{end}}) \\ &- \lambda_2 y_{\text{SF}} \log y_{\text{SF}} - \lambda_3 y_{\text{type}} \log y_{\text{type}}, \end{aligned}$$

where λ_1, λ_2, and λ_3 are weighting factors.

4 Experiments and Results

4.1 Dataset

We use the HotpotQA dataset (Yang et al., 2018), a popular benchmark for multi-hop QA task, for the main evaluation of the SRLGRN. Specifically, two sub-tasks are included in this dataset: Answer prediction and Supporting facts prediction. For each sub-task, exact match (EM) and partial match (F1) are two official evaluations that follow the work of Rajpurkar et al.. A joint EM and F1 score are used to measure the final performance of both answer and supporting fact prediction. We evaluate the model on the Distractor Setting. For each question in the Distractor Setting, two gold paragraphs and 8 distractor paragraphs, which are collected by a high-quality TF-IDF retriever from Wikipedia, are provided. Only gold paragraphs include ground-truth answers and supporting facts. In addition, we use MRC datasets, Stanford Question-Answering Dataset (SQuAD) v1.1 (Rajpurkar et al., 2016) and v2.0 (Rajpurkar et al., 2018), to demonstrate the language understanding ability of our model.

4.2 Implementation Details

We implemented SRLGRN using PyTorch[2]. We use a pre-trained BERT-base language model with 12 layers, 768-dimensional hidden size, 12 self-attention heads, and around $110M$ parameters (Devlin et al., 2018). We keep 256 words as the maximum number of words for each paragraph. For the graph construction module, we utilize a semantic role labeling model (Shi and Lin, 2019) from AllenNLP[3] to extract the predicate-argument structure. For the graph encoder module, we use 300-dimensional GloVe (Pennington et al., 2014) pre-trained word embedding. The model is optimized using Adam optimizer (Kingma and Ba, 2015).

4.3 Baselines

Baseline Model (Yang et al., 2018) makes use of Clark and Gardner approach. The model includes some neural modules that are based on self-attention and bi-attention (Seo et al., 2017).

DFGN (Xiao et al., 2019) is a strong baseline method for the HotpotQA task. DFGN builds an entity graph from the text. Moreover, DFGN includes a dynamic fusion layer that helps in finding relevant supporting facts.

SAE (Tu et al., 2019) is an effective Select, Answer and Explain system for multi-hop QA. SAE is a pipeline system that first selects the relevant paragraph and uses the selected paragraph to predict the answer and the supporting fact.

4.4 Results

Table 1 shows the results of HotpotQA (Distractor setting). We can observe the SRLGRN model exceeds most published results. Our model obtains a Joint Exact Matching (EM) score of 39.41% and Partial Matching (F1) score of 66.37% on joint performance. Our SRLGRN model has a significant improvement, about 28.58% on Joint EM and 26.21% on F1, over the Baseline Model (Yang et al., 2018). Compared to the current published state of the art, SAE model (Tu et al., 2019), our model improves EM about 2.29% and F1 about 2.56% on Answer performance and 1.41% of F1 on Joint performance. We can observe that F1 of answer span prediction is better than the current SOTA. The reason is that our model not only

[2] Our code is available at `https://github.com/HLR/SRLGRN`.

[3] `https://demo.allennlp.org/semantic-role-labeling`.

Model	Ans(%)		Sup(%)		Joint(%)	
	EM	F1	EM	F1	EM	F1
Baseline Model (Yang et al., 2018)	45.60	59.02	20.32	64.49	10.83	40.16
KGNN (Ye et al., 2019)	50.81	65.75	38.74	76.79	22.40	52.82
QFE (Nishida et al., 2019)	53.86	68.06	57.75	84.49	34.63	59.61
DecompRC (Min et al., 2019)	55.20	69.63	-	-	-	-
DFGN (Xiao et al., 2019)	56.31	69.69	51.50	81.62	33.62	59.82
TAP	58.63	71.48	46.84	82.98	32.03	61.90
SAE-base (Tu et al., 2019)	60.36	73.58	56.93	84.63	38.81	64.96
ChainEx (Chen et al., 2019)	61.20	74.11	-	-	-	-
HGN-base (Fang et al., 2019)	-	74.76	-	**86.61**	-	**66.90**
SRLGRN-base	**62.65**	**76.14**	**57.30**	85.83	**39.41**	66.37

Table 1: HotpotQA Result on Distractor setting. Except Baseline model, all models deploy BERT-base uncased as the pre-training language model to compare the performance.

uses token-level BERT representation, but also uses graph-level SRL node representations.

Our framework provides an effective way for multi-hop reasoning taking the advantages of the SRL graph model and powerful pre-trained language models. In the following section, we give a detailed analysis of the SRLGRN model.

5 Analysis

Effect of SRL Graph. The SRL graph extracts argument-predicate relationships, including in-depth semantic roles and semantic edges. The constructed graph is the basis of reasoning as the inputs of each hop are directly selected from the SRL graph, as shown in Figure 4. The SRL graph significantly improves the completeness of the graph network, that is, providing sufficient semantic edges to cover reasoning paths, see Figure 3.

Compared to the NER graph in the previous models (Xiao et al., 2019), the proposed SRL graph covers the 86.5% of complete reasoning paths for the data samples. The NER graph of DFGN is incomplete and can only cover 68.7% of the reasoning paths (Xiao et al., 2019). The graph completeness is one major reason that the SRLGRN model has higher accuracy than other published models. As shown in Table 1, the SRLGRN improves 5.79% on joint EM and 6.55% on joint F1 over DFGN, which is based on the NER graph.

Ablation	Model	Ans(%)	
		EM	F1
Graph	w/o graph	53.06	67.68
	w/o Argument type and Semantic edge	60.10	73.24
Joint	w/o joint training	58.50	71.58
Language	ALBERT-base	59.87	74.20
	BERT-base	62.65	76.14

Table 2: SRLGRN ablation study on HotpotQA.

To evaluate the effectiveness of the types of semantic roles and the edge types, we perform an ablation study. First, we removed the whole SRL graph. Second, we removed the predicate based edge information from the SRL graph. Table 2 shows the results. The complete SRLGRN improves 8.46% on F1 score compared to the model without the SRL graph. The model loses the connections used for multi-hop reasoning if we remove the SRL graph and only use BERT for answer prediction.

We also observe that the F1 score of answer span prediction decreases 2.9% if we did not incorporate semantic edge information and argument types. The reason is that removing predicate edges and argument types will destroy the argument-predicate relationships in the SRL graph and breaks the chain of reasoning. For example, in Figure 3, the main arguments of the two supporting facts in s_1^2 and s_2^2 (William and Jerry) are connected with a predicate edge, "born", to the temporal information necessary for finding the answer. Both "born" edge and the adjunct temporal roles are the key information in the two sentences to find the final answer to this question. The shared ARG node, "football player", also helps to connect the line of reasoning between the two sentences. These two results indicate that both semantic roles and semantic edges in the SRL graph are essential for the SRLGRN performance.

In a different experiment, we tested the influence of the joint training of the supporting facts and answer-prediction. As shown in Table 2, the performance will decrease by 4.56% when we did not train the model jointly.

Effect of Language Models. We use two recent and widely-used pre-trained language representation models, BERT and ALBERT (Lan et al., 2020). The last two lines of Table 2 show the results. Although BERT achieves relatively better performance, ALBERT architecture has significantly fewer parameters (18x) and is faster (about 1.7x running time) than BERT. In other words, ALBERT reduces memory consumption by cross-layer parameter sharing, increases the speed, and obtains a satisfactory performance.

Effect of SRLGRN on Single-hop QA. We evaluate the SRLGRN (excluding the paragraph selection module) on SQuAD (Rajpurkar et al., 2016) to demonstrate its reading comprehension ability. We evaluate the performance on both SQuAD v1.1 and SQuAD v2.0. Table 3 describes the comparison results with several baseline methods on SQuAD v1.1. Our model obtains a 1.8% improvement over BERT-large, and a 1.6% improvement over BERT-large+TriviaQA (Devlin et al., 2018).

Model	**Ans(%)**	
	EM	F1
Human	82.3	91.2
BERT-base	80.8	88.5
BERT-large	84.1	90.9
BERT-large+TriviaQA	84.2	91.1
BERT-large+SRLGRN	85.4	92.7

Table 3: SQuAD v1.1 performance.

We further test the SRLGRN on SQuAD v2.0. The main difference is that SQuAD v2.0 combines answerable questions (like SQuAD v1.1) with unanswerable questions (Rajpurkar et al., 2018). Table 4 shows that our proposed approach improves the performance for SQuAD benchmark compared to several recent strong baselines.

Model	**Ans(%)**	
	EM	F1
Human	86.3	89.0
ELMo+DocQA (Rajpurkar et al., 2018)	65.1	67.6
BERT-large (Devlin et al., 2018)	78.7	81.9
SemBERT (Zhang et al., 2019)	84.8	87.6
BERT-large+SRLGRN	85.8	87.9

Table 4: SQuAD v2.0 performance.

We recognize that our SRLGRN improves 7.1% on EM compared to the robust BERT-large model and improves 1.0% on EM compared to SemBERT (Zhang et al., 2019). The two experiments on SQuAD v1.1 and SQuAD v2.0 demonstrate the significance of SRL graph and the graph encoder.

Error Type	Model Prediction	Label
Synonyms	washington dc sars ey writer	district of columbia severe acute respiratory syndrome ernst young author
MLV	australian hessian mcdonald's, co	australia hessians mcdonalds
Month-Year	1946 25, november, 2015 10, july, 1873	1945 3, december 1, september, 1864
Number	11 fourth 2402	10 4 5922
External Knowledge	Coker	NCAA I FBS football
Other	taylor, swift film fourteenth	usher documentary 500th episode

Table 5: Error types on HotpotQA dev set.

6 Error Analysis

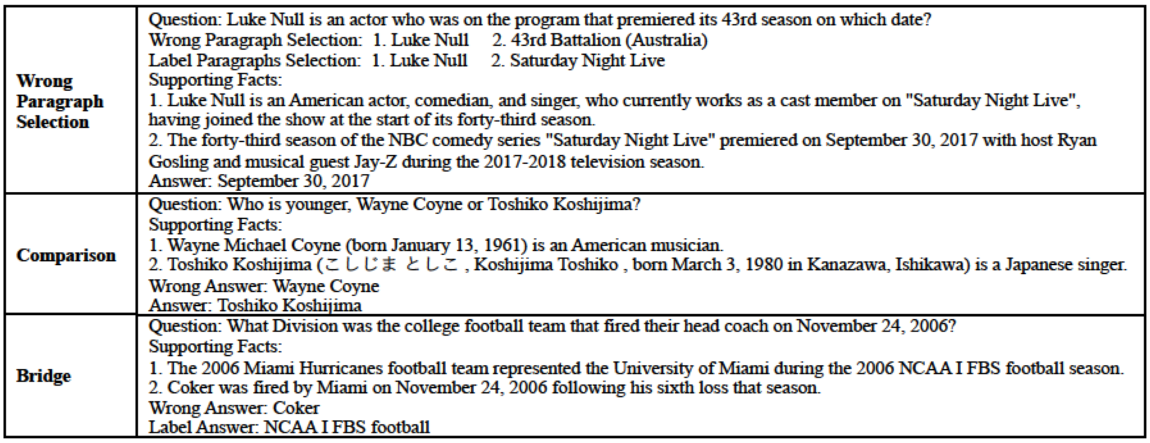
Synonyms are the most frequent cause of the reported errors in many cases where the predicted answer is semantically correct. As shown in the first row of the Table 5, our predicted answer and gold label have the same meaning. For example, SRLGRN predicts ”sars”, while the label is ”severe acute respiratory syndrome.” We know that ”sars” is the abbreviation of the gold label.

Minor Lexical Variation (MLV) is another major cause of mistakes in the SRLGRN model. As shown in the second row of Table 5, our model's predicted answer is ”australian”, while the gold label is ”australia”. Many wrong predictions occur in the singular noun versus plural noun selection.

Paragraph Selection is a small portion of errors in the SRLGRN model. As shown in Figure 5, the model chooses a wrong paragraph “43rd Battalion”. The reason is that “43rd Battalion” is a distractor although “43rd” appears in the question. The paragraph “Saturday Night Live” is the correct relevant paragraph that includes “forty-third season” and the answer. To resolve this issue in the future, we will try to combine our model with an IR system designed for multi-hopQA similar to the Multi-step entity-centric model for multi-hop QA in (Godbole et al., 2019).

Comparison and Bridge are two types of reasoning that are needed for answering HotpotQA questions. “Bridge” reasoning predicts the answer by connecting arguments to the line of reasoning that leads to the final answer. “Comparison” reasoning predicts the answer (that is, yes, no, or a text span) by comparing two arguments.

SRLGRN sometimes obtains wrong predictions in the “Comparison” reasoning when the questions



Wrong Paragraph Selection	Question: Luke Null is an actor who was on the program that premiered its 43rd season on which date? Wrong Paragraph Selection: 1. Luke Null 2. 43rd Battalion (Australia) Label Paragraphs Selection: 1. Luke Null 2. Saturday Night Live Supporting Facts: 1. Luke Null is an American actor, comedian, and singer, who currently works as a cast member on "Saturday Night Live", having joined the show at the start of its forty-third season. 2. The forty-third season of the NBC comedy series "Saturday Night Live" premiered on September 30, 2017 with host Ryan Gosling and musical guest Jay-Z during the 2017-2018 television season. Answer: September 30, 2017
Comparison	Question: Who is younger, Wayne Coyne or Toshiko Koshijima? Supporting Facts: 1. Wayne Michael Coyne (born January 13, 1961) is an American musician. 2. Toshiko Koshijima (こしじま としこ , Koshijima Toshiko , born March 3, 1980 in Kanazawa, Ishikawa) is a Japanese singer. Wrong Answer: Wayne Coyne Answer: Toshiko Koshijima
Bridge	Question: What Division was the college football team that fired their head coach on November 24, 2006? Supporting Facts: 1. The 2006 Miami Hurricanes football team represented the University of Miami during the 2006 NCAA I FBS football season. 2. Coker was fired by Miami on November 24, 2006 following his sixth loss that season. Wrong Answer: Coker Label Answer: NCAA I FBS football

Figure 5: Failing cases on our proposed SRLGRN framework.

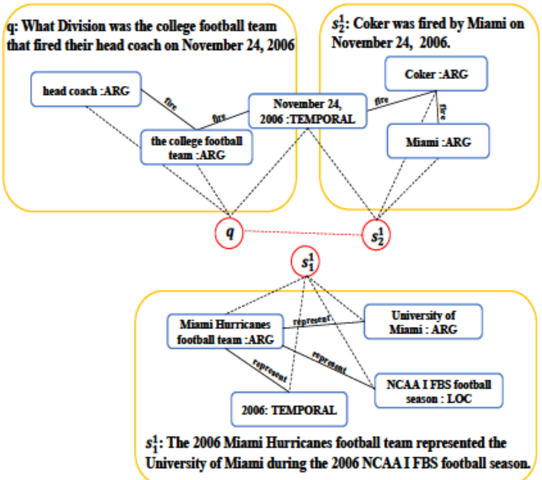

q: What Division was the college football team that fired their head coach on November 24, 2006
head coach :ARG
fire
fire
the college football team :ARG
November 24, 2006 :TEMPORAL
s_2^1: Coker was fired by Miami on November 24, 2006.
Coker :ARG
fire
fire
Miami :ARG
q
s_2^1
s_1^1
Miami Hurricanes football team :ARG
represent
University of Miami : ARG
represent
represent
NCAA I FBS football season : LOC
2006: TEMPORAL
s_1^1: The 2006 Miami Hurricanes football team represented the University of Miami during the 2006 NCAA I FBS football season.

Figure 6: The "Bridge" failing case that SRL fails to lead to the correct answer. The meaning of different lines and node colors are the same as Figure 3.

are related to "Month-year" and "Number". Our qualitative error analysis showed that SRLGRN graph leads to a wrong answer when two or more argument nodes of a same type, such as "TEMPORAL" type, are connected to one node in the graph. Moreover, We notice that the SRLGRN sometimes makes inconsistent errors. For example, in the "Comparison" failing cases of Figure 5, we predict the wrong answer "Wayne Coyne". However, we received the correct answer after replacing the word "younger" with "older".

Moreover, the "Bridge" type needs external knowledge in the HotpotQA task. As is shown in "Bridge" failing cases of Figure 5, the selected paragraphs do not show the relation between "Coker" and "Miami Hurricanes football team". Figure 6 describes the SRL construction based on this failing case. The second supporting fact and the question have the same temporal argument node "November 24, 2006". However, there is no chain between the first supporting fact and the second supporting fact due to the lack of the external knowledge that can connect "Coker", "coach" and "Miami Hurricanes football team". Therefore, the isolated reasoning chain leads to a wrong answer.

7 Conclusion

We proposed a novel semantic role labeling graph reasoning network (SRLGRN) to deal with multi-hop QA. The backbone graph of our proposed graph convolutional network (GCN) is created based on the semantic structure of the sentences. In creating the edges and nodes of the graph, we exploit a semantic role labeling sub-graph for each sentence and connect the candidate supporting facts. The cross paragraph argument-predicate structure of the sentences expressed in the graph provides an explicit representation of the reasoning path and helps in both finding and explaining the multiple hops of reasoning that lead to the final answer. SRLGRN exceeds most of the SOTA results on the HotpotQA benchmark. Moreover, we evaluate the model (excluding the paragraph selection module) on other reading comprehension benchmarks. Our approach achieves competitive performance on SQuAD v1.1 and v2.0.

Acknowledgments

This project is supported by National Science Foundation (NSF) CAREER award #1845771 and (partially) supported by the Office of Naval Research grant #N00014-19-1-2308. We thank the anonymous reviewers for their thoughtful comments.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *ICLR*.

Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, pages 31–40.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. Question answering by reasoning across documents with graph convolutional networks. In *NAACL-HLT*.

Jifan Chen, Shih-Ting Lin, and Greg Durrett. 2019. Multi-hop question answering via reasoning chains. *ArXiv*, abs/1910.02610.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *ACL*.

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

Bhuwan Dhingra, Qiao Jin, Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. 2018. Neural models for reasoning over multiple mentions using coreference. In *NAACL-HLT*.

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Güney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new qa dataset augmented with context from a search engine. *ArXiv*, abs/1704.05179.

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jing jing Liu. 2019. Hierarchical graph network for multi-hop question answering. *ArXiv*, abs/1911.03631.

Ameya Godbole, D. Kavarthapu, R. Das, Zhiyu Gong, A. Singhal, Hamed Zamani, Mo Yu, Tian Gao, Xiaoxiao Guo, M. Zaheer, and A. McCallum. 2019. Multi-step entity-centric information retrieval for multi-hop question answering. In *MRQA@EMNLP*.

Chaoyu Guan, Yuhao Cheng, and Zhao Hai. 2019. Semantic role labeling with associated memory network. In *NAACL-HLT*.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *ACL*.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *ACL*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

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

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

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *ACL*.

Diego Marcheggiani, Anton Frolov, and Ivan Titov. 2017. A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling. In *CoNLL*.

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

Kosuke Nishida, Kyosuke Nishida, Masaaki Nagata, Atsushi Otsuka, Itsumi Saito, Hisako Asano, and Junji Tomita. 2019. Answering while summarizing: Multi-task learning for multi-hop qa with evidence extraction. In *ACL*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

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*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *ICLR*.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.

Linfeng Song, Zhiguo Wang, Mo Yu, Yue Zhang, Radu Florian, and Daniel Gildea. 2018a. Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks. *ArXiv*, abs/1809.02040.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018b. A graph-to-sequence model for amr-to-text generation. In *ACL*.

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2018. Deep semantic role labeling with self-attention. In *AAAI*.

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bufang Zhou. 2019. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In *AAAI*.

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

Dirk Weissenborn, Georg Wiese, and Laura Seiffe. 2017. Making neural qa as simple as possible but not simpler. In *CoNLL*.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302.

Yunxuan Xiao, Yanru Qu, Lin Qiu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019. Dynamically fused graph network for multi-hop reasoning. In *ACL*.

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*.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *AAAI*.

D. Ye, Yankai Lin, Deming Ye, Zhenghao Liu, Z. Liu, and Maosong Sun. 2019. Multi-paragraph reasoning with knowledge-enhanced graph neural network. *ArXiv*, abs/1911.02170.

Zhuosheng Zhang, Yu-Wei Wu, Zhao Hai, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiaodong Zhou. 2019. Semantics-aware bert for language understanding. In *AAAI*.

Victor Zhong, Caiming Xiong, Nitish Shirish Keskar, and Richard Socher. 2019. Coarse-grain fine-grain coattention network for multi-evidence question answering. In *ICLR*.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *ACL*.