Propagate-Selector: Detecting Supporting Sentences for Question Answering via Graph Neural Networks

Seunghyun Yoon[1], **Franck Dernoncourt**[2], **Doo Soon Kim**[2]
Trung Bui[2] and **Kyomin Jung**[1]
[1]Dept. of ECE, Seoul National University, Seoul, Korea
[2]Adobe Research, San Jose, CA, USA
{mysmilesh,kjung}@snu.ac.kr
{franck.dernoncourt,dkim,bui}@adobe.com

Abstract

In this study, we propose a novel graph neural network, called propagate-selector (PS), which propagates information over sentences to understand information that cannot be inferred when considering sentences in isolation. First, we design a graph structure in which each node represents the individual sentences, and some pairs of nodes are selectively connected based on the text structure. Then, we develop an iterative attentive aggregation, and a skip-combine method in which a node interacts with its neighborhood nodes to accumulate the necessary information. To evaluate the performance of the proposed approaches, we conducted experiments with the HotpotQA dataset. The empirical results demonstrate the superiority of our proposed approach, which obtains the best performances compared to the widely used answer-selection models that do not consider the inter-sentential relationship.

1 Introduction

Understanding texts and being able to answer a question posed by a human is a long-standing goal in the artificial intelligence field. Given the rapid advancement of neural network-based models and the availability of large-scale datasets, such as SQuAD (Rajpurkar et al., 2016) and TriviaQA (Joshi et al., 2017), researchers have begun to concentrate on building automatic question-answering (QA) systems. One example of such a system is called the machine-reading question-answering (MRQA) model, which provides answers to questions from given passages (Xiong et al., 2016; Wang et al., 2017; Shen et al., 2017b).

Recently, research has revealed that most of the questions in the existing MRQA datasets do not require reasoning across sentences in the given context (passage); instead, they can be answered by looking at only a single sentence (Weissenborn et al., 2017). Using this characteristic, a simple model can achieve performances competitive with that of a sophisticated model. However, in most of the real scenarios of QA applications, more than one sentences should be utilized to extract a correct answer.

Passage 1, 2015 Diamond Head Classic:

① The 2015 Diamond Head Classic was a mid-season eight-team college basketball tournament... ② It was the seventh annual Diamond Head Classic tournament ... ③ No. 3-ranked Oklahoma defeated Harvard to win the tournament championship... **④** *Buddy Hield was named the tournament's MVP.*

⋮

Passage N, Buddy Hield:

① *Chavano Rainier "Buddy" Hield is a Bahamian professional basketball player for the Sacramento Kings of the NBA... ② ...*

Question: Which team does the player named 2015 Diamond Head Classic's MVP play for?

Supporting Sentences: **1-④, N-①**

Figure 1: An example of dataset. Detecting *supporting sentences* is an essential step being able to answer the question.

To alleviate this limitation in the previous datasets, another type of dataset was developed in which answering the question requires reasoning over multiple sentences in the given passages (Yang et al., 2018; Welbl et al., 2018). Figure 1 shows an example of a recently released dataset, the HotpotQA. This dataset consists of not only question-answer pairs with context passages but also *supporting sentence* information for answering the question annotated by a human.

In this study, we are interested in building a model that exploits the relational information among sentences in passages and in classifying the *supporting sentences* that contain the essential information for answering the question. To this end, we propose a novel graph neural network model, named **Propagate-selector (PS)**, that can be directly employed as a subsystem in the QA pipeline. First, we design a graph structure to hold information in the HotpotQA dataset by assigning each sentence to an independent graph node. Then, we connect the undirected edges between nodes using a proposed graph topology (see the discussion in 4.1). Next, we allow **PS** to propagate information between the nodes through iterative hops to perform reasoning across the given sentences. Trough the propagate process, the model learns to understand information that cannot be inferred when considering sentences in isolation.

To the best of our knowledge, this is the first work to employ a graph neural network structure to find *supporting sentences* for a QA system. Through experiments, we demonstrate that the proposed method achieves better performances when classifying *supporting sentences* than those of the widely used answer-selection models (Wang and Jiang, 2016; Bian et al., 2017; Shen et al., 2017a; Tran et al., 2018).

2 Related Work

Previous researchers have also investigated neural network-based models for MRQA. One line of inquiry employs an attention mechanism between tokens in the question and passage to compute the answer span from the given text (Seo et al., 2016; Wang et al., 2017). As the task scope was extended from specific- to open-domain QA, several models have been proposed to select a relevant paragraph from the text to predict the answer span (Wang et al., 2018; Clark and Gardner, 2018). However, none of these methods have addressed reasoning over multiple sentences.

To understand the relational patterns in the dataset, graph neural network algorithms have also been previously proposed. Kipf and Welling (2016) proposed a graph convolutional network to classify graph-structured data. This model was further investigated for applications involving large-scale graphs (Hamilton et al., 2017), for the effectiveness of aggregating and combining graph nodes by employing an attention mechanism (Veličković et al., 2018), and for adopting recurrent node updates (Palm et al., 2018). In addition, one trial involved applying graph neural networks to QA tasks; however, this usage was limited to the entity level rather than sentence level understanding (De Cao et al., 2018).

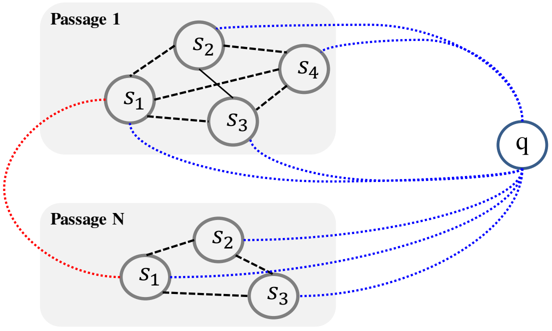


Figure 2: Topology of the proposed model. Each node represents a sentence from the passage and the question.

3 Task and Dataset

The specific problem we aim to tackle in this study is to classify *supporting sentences* in the MRQA task. We consider the target dataset HotpotQA, by Yang et al. (2018), which is comprised of tuples (<Q, P_n, Y_i, A>) where Q is the question, P_n is the set of passages as the given context, and each passage $P \in P_n$ is further comprised of a set of sentences S_i ($S_i \in P_n$). Here, Y_i is a binary label indicating whether S_i contains the information required to answer the question, and A is the answer. In particular, we call a sentence, $S_s \in S_i$, a *supporting sentence* when Y_s is *true*. Figure 1 shows an example of the HotpotQA dataset.

In this study, we do not use the answer information from the dataset; we use only the subsequent tuples <Q, P_n, Y_i> when classifying *supporting sentences*. We believe that this subproblem plays an important role in building a full QA pipeline because the proposed models for this task will be combined with other MRQA models in an end-to-end training process.

4 Methodology

4.1 Propagate-Selector

In this paper, we are interested in identifying *supporting sentences*, among sentences in the given text that contain information essential to answering the question. To build a model that can perform reasoning across multiple sentences, we propose a graph neural network model called **Propagate-selector** (**PS**). **PS** consists of the following parts:

Topology: To build a model that understands the relationship between sentences for answering a question, we propose a graph neural network where each node represents a sentence from passages and the question. Figure 2 depicts the topology of the proposed model. In an offline step, we organize the content of each instance in a graph where each node represents a sentence from the passages and the question. Then, we add edges between nodes using the following topology:

- we fully connect nodes that represent sentences from the same passage (dotted-black);
- we fully connect nodes that represent the first sentence of each passage (dotted-red);
- we add an edge between the question and every node for each passage (dotted-blue).

In this way, we enable a path by which sentence nodes can propagate information between both inner and outer passages.

Node representation: Question $\mathbf{Q} \in \mathbb{R}^{d \times Q}$ and sentence $\mathbf{S}_i \in \mathbb{R}^{d \times S_i}$, (where d is the dimensionality of the word embedding and Q and S_i represent the lengths of the sequences in $\mathbf{Q}$ and $\mathbf{S}_i$, respectively), are processed to acquire the sentence-level information. Recent studies have shown that a pretrained language model helps the model capture the contextual meaning of words in the sentence (Peters et al., 2018; Devlin et al., 2019). Following this study, we select an ELMo (Peters et al., 2018) language model for the word-embedding layer of our model as follows: $\mathbf{L}^Q = \text{ELMo}(\mathbf{Q})$, $\mathbf{L}^S = \text{ELMo}(\mathbf{S})$. Using these new representations, we compute the sentence representation as follows:

$$\begin{aligned} \mathbf{h}_t^Q &= f_\theta(\mathbf{h}_{t-1}^Q, \mathbf{L}_t^Q), \\ \mathbf{h}_t^S &= f_\theta(\mathbf{h}_{t-1}^S, \mathbf{L}_t^S), \\ \mathbf{N}^Q &= \mathbf{h}_{\text{last}}^Q, \;\; \mathbf{N}^S = \mathbf{h}_{\text{last}}^S, \end{aligned} \tag{1}$$

where f_θ is the RNN function with the weight parameters θ, and $\mathbf{N}^Q \in \mathbb{R}^{d'}$ and $\mathbf{N}^S \in \mathbb{R}^{d'}$ are node representations for the question and sentence, respectively (where d' is the dimensionality of the RNN hidden units).

Aggregation: An iterative attentive aggregation function to the neighbor nodes is utilized to compute the amount of information to be propagated to each node in the graph as follows:

$$\begin{aligned} \mathbf{A}_v^{(k)} &= \sigma(\textstyle\sum_{u \in N(v)} a_{vu}^{(k)} \, \mathbf{W}^{(k)} \cdot \mathbf{N}_u^{(k)}), \\ a_{vu}^{(k)} &= \exp(\mathbf{S}_{vu}) / \textstyle\sum_k \exp(\mathbf{S}_{vk}), \\ \mathbf{S}_{vu}^{(k)} &= (\mathbf{N}_v^{(k)})^\intercal \cdot \mathbf{W}^{(k)} \cdot \mathbf{N}_u^{(k)}, \end{aligned} \tag{2}$$

where $\mathbf{A}_v \in \mathbb{R}^{d'}$ is the aggregated information for the v-th node computed by attentive weighted summation of its neighbor nodes, a_{vu} is attention weight between node v and its neighbor nodes u $(u \in N(v))$, $\mathbf{N}_u \in \mathbb{R}^{d'}$ is the u-th node representation, σ is a nonlinear activation function, and $\mathbf{W} \in \mathbb{R}^{d' \times d'}$ is the learned model parameter. Because all the nodes belong to a graph structure in which the iterative aggregation is performed among nodes, the k in the equation indicates that the computation occurs in the k-th hop (iteration).

Update: The aggregated information for the v-th node, $\mathbf{A}_v$ in equation (2), is combined with its previous node representation to update the node. We apply a skip connection to allow the model to learn the amount of information to be updated in each hop as follows:

$$\mathbf{N}_v^{(k)} = \sigma(\mathbf{W} \cdot \{\mathbf{N}_v^{(k-1)}; \mathbf{A}_v^{(k)}\}), \tag{3}$$

where σ is a nonlinear activation function, $\{;\}$ indicates vector concatenation, and $\mathbf{W} \in \mathbb{R}^{d' \times 2d'}$ is the learned model parameter.

4.2 Optimization

Because our objective is to classify *supporting sentences* ($S_i \in P_n$) from the given tuples $<Q$, P_n, $Y_i>$, we define two types of loss to be minimized. One is a rank loss that computes the cross-entropy loss between a question and each sentence using the ground-truth Y_i as follows:

$$\begin{aligned} \text{loss}_{rank} &= -\log \textstyle\sum_{i=1}^{N} Y_i \log(S_i), \\ \text{S} &= [\text{score}_1, ..., \text{score}_i], \\ \text{score}_i &= g_\theta(\mathbf{N}^Q, \mathbf{N}_i^S), \end{aligned} \tag{4}$$

where g_θ is a feedforward network that computes a similarity score between the final representation of the question and each sentence. The other is attention loss, which is defined in each hop as follows:

$$\text{loss}_{attn} = -\log \textstyle\sum_{i=1}^{k} \sum_{i=1}^{N} Y_i \log(a_{qi}^{(k)}), \tag{5}$$

where $a_{qi}^{(k)}$ indicates the relevance between the question node q and the i-th sentence node in the k-th hop as computed by equation (2).

Finally, these two losses are combined to construct the final objective function:

$$\mathcal{L} = \alpha \; \text{loss}_{rank} + \text{loss}_{attn}, \tag{6}$$

where α is a hyperparameter.

properties	**train**	**dev**
# questions	90,447	7,405
# sentences	3,703,344	306,487
passages / question	9.95	9.95
sentences / passage	4.12	4.16
sentences / question	40.94	41.39
supporting sentences / question	2.39	2.43
avg tokens (question)	17.92	15.83
avg tokens (sentence)	22.38	22.41

Table 1: Properties of the dataset.

5 Experiments

We regard the task as the problem of selecting the *supporting sentences* from the passages to answer the questions. Similar to the answer-selection task in the QA literature, we report the model performance using the mean average precision (MAP) and mean reciprocal rank (MRR) metrics. To evaluate the model performance, we use the HotpotQA dataset, which is described in section "Task and Dataset". Table 1 shows properties of the dataset. We conduct a series of experiments to compare baseline methods with the newly proposed models. All codes developed for this research will be made available via a public web repository along with the dataset.

5.1 Implementation Details

To implement the **Propagate-selector** (**PS**) model, we first use a small version of ELMo (13.6 *M* parameters) that provides 256-dimensional context embedding. This choice was based on the available batch size (50 for our experiments) when training the complete model on a single GPU (GTX 1080 Ti). When we tried using the original version of ELMo (93.6 *M* parameters, 1024-dimensional context embedding), we were able to increase the batch size only up to 20, which results in excessive training time (approximately 90 hours). For the sentence encoding, we used a GRU (Chung et al., 2014) with a hidden unit dimension of 200. The hidden unit weight matrix of the GRU is initialized using orthogonal weights (Saxe et al., 2013). Dropout is applied for regularization purposes at a ratio of 0.7 for the RNN (in equation 1) to 0.7 for the attention weight matrix (in equation 2). For the nonlinear activation function (in equation 2 and 3), we use the $tanh$ function.

Regarding the vocabulary, we replaced vocabulary with fewer than 12 instances in terms of term-frequency with "*UNK*" tokens. The final vocabulary size was 138,156. We also applied the Adam optimizer (Kingma and Ba, 2014), including gradient clipping by norm at a threshold of 5.

Model	dev		train	
	MAP	MRR	MAP	MRR
IWAN [1]	0.526	0.680	0.605	0.775
sCARNN [2]	0.534	0.698	0.620	0.792
CompAggr [3]	0.659	0.812	0.796	0.911
CompClip [4]	0.670	0.825	0.767	0.901
CompClip-LM [5]	0.696	0.841	0.748	0.873
PS-*avg*	0.566	0.708	0.889	0.959
PS-*rnn*	0.700	0.822	**0.919**	**0.971**
PS-*rnn-elmo-s*	0.716	0.841	0.813	0.916
PS-*rnn-elmo*	**0.734**	**0.853**	0.863	0.945

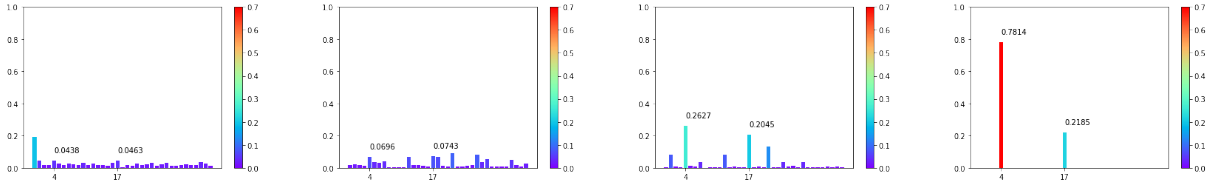
Table 2: Model performance on the HotpotQA dataset (top scores marked in bold). Models [1-5] are from (Shen et al., 2017a; Tran et al., 2018; Wang and Jiang, 2016; Bian et al., 2017; Yoon et al., 2019), respectively.

5.2 Comparisons with Other Methods

Table 2 shows the model performances on the HotpotQA dataset. Because the dataset only provides training (trainset) and validation (devset) subsets, we report the model performances on these datasets. While training the model, we implement early termination based on the devset performance and measure the best performance. To compare the model performances, we choose widely used answer-selection models such as **CompAggr** (Wang and Jiang, 2016), **IWAN** (Shen et al., 2017a), **CompClip** (Bian et al., 2017), **sCARNN** (Tran et al., 2018), and **CompClip-LM** (Yoon et al., 2019) which were primarily developed to rank candidate answers for a given question. The **CompClip-LM** is based on **CompClip** and adopts ELMo in its word-embedding layer.

In addition to the main proposed model, **PS**-*rnn-elmo*, we also investigate three model variants: **PS**-*rnn-elmo-s* uses a small version of ELMo, **PS**-*rnn* uses GloVe (Pennington et al., 2014) instead of ELMo as a word-embedding layer, and **PS**-*avg* employs average pooling ($\mathbf{N}^Q$= average($\mathbf{Q}$) and $\mathbf{N}^S$= average($\mathbf{S}$)) instead of RNN encoding in equation (1).

As shown in Table 2, the proposed **PS**-*rnn-elmo* shows a significant MAP performance im-



(a) hop-1 (b) hop-2 (c) hop-3 (d) hop-4

Figure 3: Attention weights between the question and sentences in the passages. As the number of hops increases, the proposed model correctly classifies *supporting sentences* (ground-truth index 4 and 17).

# hop	dev		train	
	MAP	MRR	MAP	MRR
1	0.651	0.794	0.716	0.842
2	0.653	0.797	0.721	0.850
3	0.698	0.830	0.800	0.908
4	**0.734**	**0.853**	**0.863**	**0.945**
5	0.700	0.827	0.803	0.906
6	0.457	0.606	0.467	0.621

Table 3: Model performance (top scores marked in bold) as the number of hop increases.

# hop	dev		train	
	MAP	MRR	MAP	MRR
1	0.648	0.790	0.708	0.842
2	0.655	0.801	0.720	0.853
3	0.681	0.816	0.768	0.886
4	0.706	0.834	0.796	0.906
5	**0.716**	**0.841**	**0.813**	**0.916**
6	0.441	0.596	0.452	0.600
7	0.434	0.589	0.450	0.606

Table 4: Model performance with small version of ELMo (top scores marked in bold) as the number of hop increases.

provement compared to the previous best model, **CompClip-LM** (0.696 to 0.734 absolute).

5.3 Hop Analysis

Table 3 shows the model performance (**PS**-*elmo*) as the number of hops increases. We find that the model achieves the best performance in the 4-hop case but starts to degrade when the number of hops exceeds 4. We assume that the model experiences the vanishing gradient problem under a larger number of iterative propagations (hops). Table 4 shows model performance with small version of ELMo.

Figure 3 depicts the attention weight between the question node and each sentence node (hop-4 model case). As the hop number increases, we observe that the model properly identifies *supporting sentences* (in this example, sentence #4 and #17). This behavior demonstrates that our proposed model correctly learns how to propagate the necessary information among the sentence nodes via the iterative process.

6 Conclusion

In this paper, we propose a graph neural network that finds the sentences crucial for answering a question. The experiments demonstrate that the model correctly classifies *supporting sentences* by iteratively propagating the necessary information through its novel architecture. We believe that our approach will play an important role in building a QA pipeline in combination with other MRQA models trained in an end-to-end manner.

References

Weijie Bian, Si Li, Zhao Yang, Guang Chen, and Zhiqing Lin. 2017. A compare-aggregate model with dynamic-clip attention for answer selection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1987–1990. ACM.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 845–855.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2018. Question answering by reasoning across documents with graph convolutional networks. *arXiv preprint arXiv:1808.09920*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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*, pages 4171–4186.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1601–1611.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Rasmus Palm, Ulrich Paquet, and Ole Winther. 2018. Recurrent relational networks. In *Advances in Neural Information Processing Systems*, pages 3368–3378.

Jeffrey Pennington, Richard Socher, and Christopher 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.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Andrew M Saxe, James L McClelland, and Surya Ganguli. 2013. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. In *International Conference on Learning Representations (ICLR)*.

Gehui Shen, Yunlun Yang, and Zhi-Hong Deng. 2017a. Inter-weighted alignment network for sentence pair modeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1179–1189.

Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. 2017b. Reasonet: Learning to stop reading in machine comprehension. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1047–1055. ACM.

Quan Hung Tran, Tuan Lai, Gholamreza Haffari, Ingrid Zukerman, Trung Bui, and Hung Bui. 2018. The context-dependent additive recurrent neural net. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 1274–1283.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations (ICLR)*.

Shuohang Wang and Jing Jiang. 2016. A compare-aggregate model for matching text sequences. *arXiv preprint arXiv:1611.01747*.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R 3: Reinforced ranker-reader for open-domain question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 189–198.

Dirk Weissenborn, Georg Wiese, and Laura Seiffe. 2017. Making neural qa as simple as possible but not simpler. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 271–280.

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

Caiming Xiong, Victor Zhong, and Richard Socher. 2016. Dynamic coattention networks for question answering. *arXiv preprint arXiv:1611.01604*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

Seunghyun Yoon, Franck Dernoncourt, Doo Soon Kim, Trung Bui, and Kyomin Jung. 2019. A compare-aggregate model with latent clustering for answer selection. *arXiv preprint arXiv:1905.12897*.