The Role of RNNs for Contextual Representations: A Case Study Using DMN+

Yuanyuan Shen
Department of Information Technology and Software Engineering
Auckland University of Technology
Auckland, New Zealand
yuanyuan.shen@aut.ac.nz

Edmund M-K Lai
Department of Information Technology and Software Engineering
Auckland University of Technology
Auckland, New Zealand
edmund.lai@aut.ac.nz

Mahsa Mohaghegh
Department of Information Technology and Software Engineering
Auckland University of Technology
Auckland, New Zealand
mahsa.mohaghegh@aut.ac.nz

ABSTRACT

Recurrent neural networks (RNNs) have been used prevalently to capture long-term dependencies of sequential inputs. In particular, for question answering systems, variants of RNNs, such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), allow the positional, ordering or contextual information to be encoded into latent contextual representations. While applying RNNs for encoding this information is intuitively reasonable, no specific research has been conducted to investigate how effective is their use in such systems when the sequence of sentences is unimportant. In this paper we conduct a case study on the effectiveness of using RNNs to generate context representations using the DMN+ network. Our results based on a three-fact task in the bAbI dataset show that sequences of facts in the training dataset influence the predictive performance of the trained system. We propose two methods to resolve this problem, one is data augmentation and the other is the optimization of the DMN+ structure by replacing the GRU in the episodic memory module with a non-recurrent operation. The experimental results demonstrate that our proposed solutions can resolve the problem effectively.

CCS CONCEPTS

•Computing methodologies → Machine learning → Machine learning approaches → Neural networks •Computing methodologies → Artificial intelligence → Natural language processing → Information extraction

KEYWORDS

Question Answering, Recurrent Neural Networks, Contextual Representations, Deep Learning

CLNLP'20, July, 2020, Seoul, South Korea

https://doi.org/10.1145/1234567890

ACM Reference format:

Yuanyuan Shen, Edmund M-K Lai and Mahsa Mohaghegh. 2020. The Role of RNNs for Contextual Representations: A Case Study Using DMN+. In *Proceedings of 2020 International Conference on Computational Linguistics and Natural Language Processing (CLNLP'20). Seoul, South Korea, 6 pages.* https://doi.org/10.1145/1234567890

1 Introduction

In recent years, with the advance of deep learning, a substantial amount of research efforts has been devoted to applying it to question answering (QA), one of the oldest problems in natural language processing (NLP). QA can be used to retrieve or infer answers for question posted by users, and it can be incorporated into dialog systems and chatbots. The main advantage of deep learning methods is that they do not require any feature-engineering. A review of the research in QA systems can be found in [1, 2].

Most of the neural network-based QA systems contains one or more recurrent neural networks (RNNs) because they are able to capture sequential dependencies of data. In a basic RNN, the state of the hidden node is fed back into itself [3]. In principle, a large enough RNN is capable of learning sequences of arbitrary length. In practice, however, standard RNNs are unable to learn a very long sequence of tokens because of a numerical problem known as the vanishing or exploding gradient problem during training. The use of Long Short-Term Memory (LSTM) [4] and Gated Recurrent Unit (GRU) [5] has largely resolved this problem and therefore they are used in most, if not all, networks that require RNN nowadays.

One of the earliest such QA systems is MemNN proposed in 2015 [6]. It employs an RNN for one of the core modules to predict the textual responses with being fed the sequence of question and supporting memories. Shortly later an end-to-end version of MemNN was proposed in [7] called MemN2N. In each hop, the attention weighted sentence representations are passed into an RNN to generate an internal context vector. The application of the RNN in MemN2N plays an important role of inspiring a number of neural network-based end-to-end QA systems, including

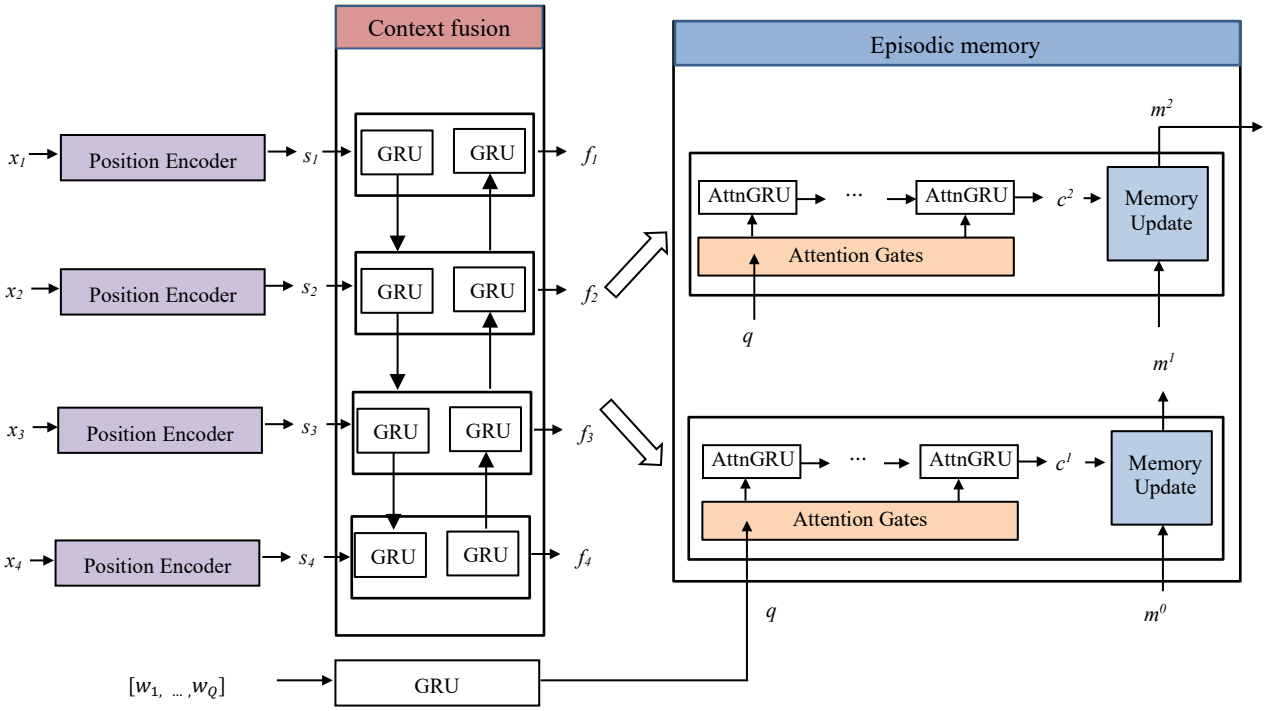

Figure 1: Block diagram of DMN+ QA System

dynamic memory networks (DMN) [8] and DMN+ [9]. DMN uses GRU to encode the input sentences to generate the final states as the sentence vectors. These vectors form the context memory which is then weighted by attention gates which are themselves GRU networks. In DMN+, bi-directional GRU is leveraged to encode the context from both preceding and succeeding directions for sentence representation. More recently, R-Net [10], BiDAF [11], DCN [12] and FastQA [13] are end-to-end QA systems that produce state-of-the-art performances. All these systems make comprehensive use of RNNs to encode input sequence and to implement the attention mechanisms.

While the use of RNNs is important in QA tasks where the order of sentences is significant, to the best of our knowledge, no research has been conducted to study the impact they have for the tasks where the order of sentences is irrelevant. In this paper, the results of such a study is reported. Here, we use DMN+ on the bAbI dataset [14]. We demonstrate that the sequences of facts in the training dataset influence the predictive performance of the trained system. We propose two methods to overcome this problem. The first one is data augmentation. The second one is to replace the GRU in the episodic memory module. Our results showed that both methods are effective.

2 The DMN+ QA System

The architecture of the DMN+ system is illustrated in Figure 1. The words are converted into the high dimensional space using word embeddings $\{\boldsymbol{w}_i\}_{i=1}^{V} \in \mathbb{R}^d$, where d denotes the dimension of the word embeddings and v is the size of the vocabulary. The question vector q is the final sate of the GRU fed by the sequence of word embeddings in the question. The K word embeddings within a sentence are stacked to form the initial sentence representations $\{\boldsymbol{x}_i\}_{i=1}^{N} \in \mathbb{R}^{K\times d}$, which then are encoded into the sentence vectors $\{\boldsymbol{s}_i\}_{i=1}^{N} \in \mathbb{R}^d$ by the Position Encoder (PE) [7] in order to capture the positional information of each word within a sentence. More specifically the PE are a weighting matrix $PE \in \mathbb{R}^{K\times d}$. The sentence vector is calculated by

$$s_i = \sum_{i=1}^{K} PE \cdot x_i \tag{1}$$

where $\cdot$ is the element-wise multiplication.

These positional encoded sentence vectors are fed into a bi-directional GRU module which serves as the context fusion layer.

$$\overrightarrow{h_i} = \overrightarrow{GRU}(h_{i-1}, s_i) \tag{2}$$

$$\overleftarrow{h_i} = \overleftarrow{GRU}(h_{i+1}, s_i) \tag{3}$$

$$f_i = \overrightarrow{h_i} + \overleftarrow{h_i} \tag{4}$$

The output of this layer $\{\boldsymbol{f}_i\} \in \mathbb{R}^d$ forms the input to the episodic memory. The episodic memory module consists of multiple hops to retrieve information from sentence representations by paying attention to a subset of sentences. Each hop contains the following components: attention mechanism, attention based GRU to generate the

context representations and the episodic memory vector update for output to the next hop.

The attention mechanism takes the sentence representation, question vector and the current memory vector as the input and computes the attention gate $g_i^t \in \mathbb{R}$ for the sentence f_i:

$$\alpha_i^t = tanh\left(W^{(1)} z_i^t(f_i, m^t, q) + b^{(1)}\right) \quad (5)$$

$$g_i^t = softmax(W^{(2)} \alpha_i^t + b^{(2)}) \quad (6)$$

where $z_i^t(s_i, m^t, q) = [f_i \cdot q;\ f_i \cdot m^t;\ |f_i - q|;\ |f_i - m^t|] \in \mathbb{R}^{4d}$. m^t is the memory vector in hop t. $|\cdot|$ denotes the absolute value.

In the attention based GRU, the update gate in a standard GRU is replaced by attention gates as follows:

$$r_i = sigmoid(W^{(r)} f_i + U^{(r)} h_{i-1} + b^{(u)}) \quad (7)$$

$$\widetilde{h_i} = tanh(W f_i + r_i U h_{i-1} + b) \quad (8)$$

$$h_i = g_i^t \widetilde{h_i} + (1 - g_i^t) h_{i-1} \quad (9)$$

In this way, the positional and ordering information of the sentences as well as those sentences with relatively higher attention weights are preserved in the final hidden state of the attention based GRU, which acts as the context vector c^t.

This context representation is then passed to the memory update component which is a feedforward neural network with ReLU activation. The episodic memory vector of the last hop is used to predict the answer:

$$\hat{a} = softmax(W^{(a)} m^T) \quad (10)$$

where T is the number of hops.

3 Effects of Changing the Order of Facts

We shall investigate the effects of changing the ordering of facts on the performance of the DMN+ system using the bAbI en-10k dataset. This dataset is one of the benchmarks for QA systems. It consists of 20 tasks, each requiring different reasoning abilities to answer the questions. We shall focus on Task 16 which is a three-fact reasoning task. In other words, the answer is derived from three supporting sentences in the story. For the data in this task, manipulating the order of the supporting facts does not affect the answer. There are therefore 6 possible permutations of the order of the three supporting facts, as shown in Table 1. F1, F2 and F3 means the first fact needs to be found, the second and the third. Samples of the 6 patterns are shown in Figure 2. We categorize the training set into these 6 order patterns. There are a total of 1666 samples

Table 1: Possible orders of 3 supporting facts.

Order patterns	Orders of facts
1	F1, F2, F3
2	F1, F3, F2
3	F2, F1, F3
4	F2, F3, F1
5	F3, F1, F2
6	F3, F2, F1

1 Lily is a frog. 2 Bernhard is a lion. 3 Bernhard is gray. 4 Julius is a swan. 5 Brian is a frog. 6 Greg is a swan. 7 Julius is green. 8 Brian is yellow. 9 Greg is green. 10 What color is Lily? yellow 1 5 8 (a)	1 Greg is a frog. 2 Greg is green. 3 Julius is a rhino. 4 Lily is a swan. 5 Bernhard is white. 6 Lily is gray. 7 Bernhard is a rhino. 8 Brian is a lion. 9 Brian is yellow. 10 What color is Julius? white 3 7 5 (b)	1 Brian is a swan. 2 Julius is a rhino. 3 Greg is a swan. 4 Lily is a rhino. 5 Lily is gray. 6 Julius is gray. 7 Brian is white. 8 Bernhard is a lion. 9 Bernhard is yellow. 10 What color is Greg? white 3 1 7 (c)
1 Bernhard is white. 2 Julius is a swan. 3 Bernhard is a swan. 4 Brian is a frog. 5 Brian is gray. 6 Lily is a lion. 7 Greg is a rhino. 8 Lily is yellow. 9 Greg is yellow. 10 What color is Julius? white 2 3 1 (d)	1 Bernhard is a swan. 2 Bernhard is white. 3 Julius is a lion. 4 Greg is a rhino. 5 Greg is white. 6 Lily is a lion. 7 Julius is green. 8 Lily is green. 9 Brian is a swan. 10 What color is Brian? white 9 1 2 (e)	1 Julius is a lion. 2 Bernhard is a rhino. 3 Brian is yellow. 4 Julius is green. 5 Brian is a swan. 6 Bernhard is gray. 7 Lily is a lion. 8 Lily is white. 9 Greg is a swan. 10 What color is Greg? yellow 9 5 3 (f)

Figure 2: Samples of the 6 patterns. (a) a sample of pattern 1; (b) a sample of pattern 2; (c) a sample of pattern 3; (d) a sample of pattern 4; (e) a sample of pattern 5; (f) a sample of pattern 6.

Table 2: The test accuracy (%) on each pattern of DMN+ trained on each pattern

Test pattern	Order of facts	Trained Pattern					
		1	2	3	4	5	6
1	F1, F2, F3	73.3	43.3	65.7	34.7	43.7	38.0
2	F1, F3, F2	51.3	68.0	55.0	55.7	67.3	51.7
3	F2, F1, F3	73.0	46.0	78.3	38.3	41.7	37.0
4	F2, F3, F1	38.3	44.7	30.0	70.0	42.0	67.7
5	F3, F1, F2	54.7	67.3	46.7	53.3	64.7	48.7
6	F3, F2, F1	36.0	45.3	28.7	69.3	42.7	72.0

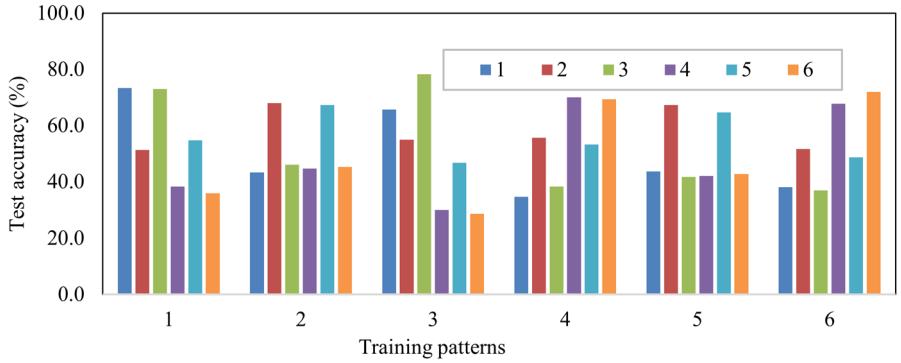

Figure 3: Graphical presentation of the results in Table 2

with each pattern, which are then divided into the training, validation and testing sets. We use 20% for the testing set and the rest samples are split with 9:1 for the training and validation sets.

3.1 Experimental Results

A separate DMN+ network is trained with the samples of each of the 6 patterns respectively, resulting in 6 different trained models. Each of these 6 models is then tested with the test samples of all the 6 patterns. The test accuracies are shown in Table 2 and plotted graphically in Figure 3.

It is obvious that the accuracy of each trained network is highest for the pattern in which they are trained. For example, the highest accuracy for Network 1 is 73.3% for the test samples with pattern 1. But its accuracy is significantly lower for patterns 2, 4, and 6. This trend is the same for the other five trained networks. This shows that the trained models memorized the sequence of facts in the training dataset, and they do not generalize well to other orders of facts which are not present in the training data.

4 Possible Solutions

Two possible solutions to resolve the issue presented in the previous section are explored here. The first one is to enhance the training through data augmentation. The second is by a modification to the episodic memory module of DMN+.

4.1 Data Augmentation

Since the original training data may not present a full complement of the ordering of facts, a well-known technique in artificial neural networks is to supplement these data with augmented data. In this context, augmentation involves permutating the order of the sentences in the original dataset, keeping the answer unchanged. The network will then be trained by this larger, augmented dataset.

A network is trained on this augmented dataset and tested on each pattern. The results are shown in Table 3. For convenience, the results in Table 1 are also included in this table. Figure 3 is a graphical presentation of the results in Table 3.

These results show that the network trained with augmented data performed more or less equally for all test patterns. This clearly demonstrates that the network memorizes the order of the facts in the training samples. That is why the network trained with augmented data is not able to achieve the level of accuracies if it were trained and tested with the same patterns.

Table 3: Test accuracy (%) of MoDMN+ on each pattern

Test pattern	Order of facts	Trained Pattern						
		1	2	3	4	5	6	Augmented Data
1	F1, F2, F3	73.3	43.3	65.7	34.7	43.7	38.0	54.3
2	F1, F3, F2	51.3	68.0	55.0	55.7	67.3	51.7	62.0
3	F2, F1, F3	73.0	46.0	78.3	38.3	41.7	37.0	59.3
4	F2, F3, F1	38.3	44.7	30.0	70.0	42.0	67.7	56.3
5	F3, F1, F2	54.7	67.3	46.7	53.3	64.7	48.7	62.0
6	F3, F2, F1	36.0	45.3	28.7	69.3	42.7	72.0	54.0

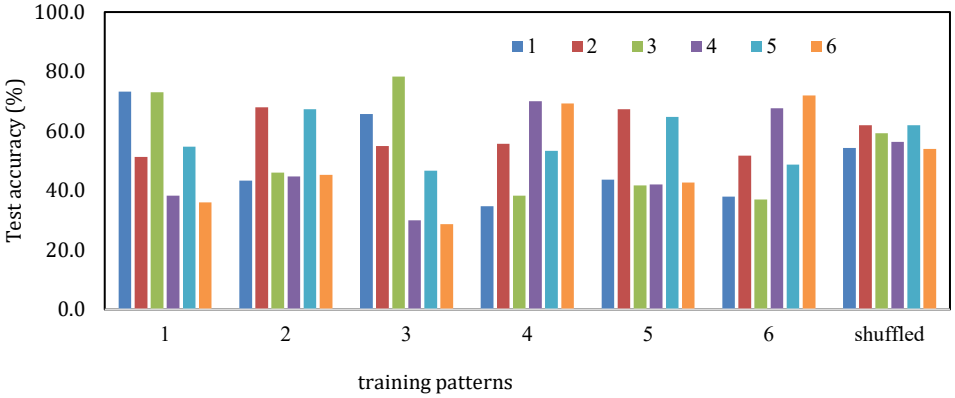

Figure 4: Graphical presentation of the results in Table 3.

4.2 Modifying the Episodic Memory Structure

The phenomenon introduced in Section 3.1 is mainly due to the RNN, in the form of GRU, in the episodic memory module. This is illustrated by replacing the attention-based GRU in the original DMN+ structure by attention-weighted summation of sentence vectors. This new structure is in the same spirit as soft attention that was mentioned in [9]. We shall refer to the modified DMN+ as MoDMN+. In this modified network, the information related to the order of sentences is not incorporated into the contextual vector. Consequently, it should have no influence on the predictive performance with different test patterns.

Six MoDMN+ networks are trained in the same way as in Section 3.1. The results for the 6 test patterns are shown in Table 3 and graphically illustrated in Figure 4.

Comparing to the results shown in Table 2, the large variations in test accuracies are significantly reduced. For example, for Network 1, the accuracies for test patterns 4 and 6 are now increased to 53% and 50% respectively, compared with 38.3% and 36% in Table 2. This suggests that the GRU in the episodic memory module does play an important role in this fact-order phenomenon. Replacing these GRU with a non-recurrent calculation does help to improve the generalization capability of the model for the tasks where the order of facts is not relevant. It is worth noting that the highest accuracies are achieved by Networks 2 and 5 when tested on patterns 2 and 5.

5 Conclusions

In this paper, by using DMN+ we have shown that applying RNNs to preserve the positional and ordering information of facts in neural network-based QA systems could cause substantial performance degradation in some circumstances. This is caused by the RNNs memorizing the order of facts in the training dataset. When these trained networks are presented with test cases where the order of facts is not adequately present in the training data, they performed poorly. We showed that this problem can be alleviated by training data augmentation without changing the original model. We have also proposed an attention-weighted summation of sentence representations to replace the GRU in the episodic memory module. This has also shown to alleviate the problem. Future work will explore the new structure on a larger number of facts and the influence of one pattern on another.

Table 4: The test accuracy (%) on each pattern of DMN+ trained on different patterns

Test pattern	Order of facts	Trained pattern					
		1	2	3	4	5	6
1	F1, F2, F3	61.0	52.0	54.3	52.7	54.7	51.0
2	F1, F3, F2	48.3	66.3	48.7	48.3	65.3	45.0
3	F2, F1, F3	53.7	49.3	60.3	55.7	51.7	50.3
4	F2, F3, F1	53.0	49.0	60.0	53.0	56.0	53.0
5	F3, F1, F2	51.7	69.0	52.7	51.7	64.7	44.3
6	F3, F2, F1	50.0	49.7	53.3	52.0	43.0	56.3

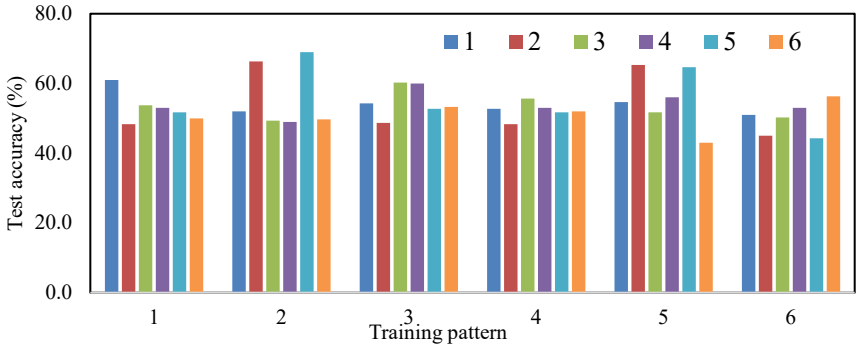

Figure 5: Graphical presentation of the results in Table 4.

ACKNOWLEDGMENTS

The authors would like to thank Callaghan Innovation for the R&D Fellowship Grants and Ambit AI Limited for the research sponsorship.

REFERENCES

[1] Y. Sharma and S. Gupta, "Deep Learning Approaches for Question Answering System," *Procedia computer science,* vol. 132, pp. 785-794, 2018.

[2] K. Ishwari, A. Aneeze, S. Sudheesan, H. Karunaratne, A. Nugaliyadde, and Y. Mallawarrachchi, "Advances in Natural Language Question Answering: A Review," *arXiv preprint arXiv:1904.05276,* 2019.

[3] F. J. Pineda, "Generalization of back-propagation to recurrent neuralnetworks," *Physical review letters,* vol. 59, p. 2229, 1987.

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation,* vol. 9, pp. 1735-1780, 1997.

[5] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, *et al.*, "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724-1734.

[6] J. Weston, S. Chopra, and A. Bordes, "Memory networks," in *International Conference on Learning Representations (ICLR)*, 2015.

[7] S. Sukhbaatar, J. Weston, and R. Fergus, "End-to-end memory networks," in *Advances in neural information processing systems (NIPS)*, 2015, pp. 2440-2448.

[8] A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, *et al.*, "Ask me anything: Dynamic memory networks for natural language processing," in *International Conference on Machine Learning (ICML)*, 2016, pp. 1378-1387.

[9] C. Xiong, S. Merity, and R. Socher, "Dynamic memory networks for visual and textual question answering," in *International Conference on Machine Learning (ICML)*, 2016, pp. 2397-2406.

[10] W. Wang, N. Yang, F. Wei, B. Chang, and M. Zhou, "Gated self-matching networks for reading comprehension and question answering," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 189-198.

[11] M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," in *International Conference on Learning Representations (ICLR)*, 2018.

[12] C. Xiong, V. Zhong, and R. Socher, "Dynamic coattention networks for question answering," presented at the International Conference on Learning Representations (ICLR), 2017.

[13] D. Weissenborn, G. Wiese, and L. Seiffe, "Making Neural QA as Simple as Possible but not Simpler," in *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, 2017, pp. 271-280.

[14] J. Weston, A. Bordes, S. Chopra, A. M. Rush, B. van Merriënboer, A. Joulin, *et al.*, "Towards ai-complete question answering: A set of prerequisite toy tasks," in *International Conference on Learning Representations (ICLR)*, 2016.