Attention-Based Memory Network for Sentence-Level Question Answering

Pei Liu[1(✉)], Chunhong Zhang[1], Weiming Zhang[1], Zhiqiang Zhan[2], and Benhui Zhuang[1]

[1] Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing, China
{peiliu, zhangch, zhangwm, zhuangbenhui}@bupt.edu.cn
[2] School of Information and Communication Engineering, Engineering Research Center of Information Networks, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing, China
zqzhan@bupt.edu.cn

Abstract. Sentence-level question answering (QA) for news articles is a promising task for social media, whose task is to make machine understand a news article and answer a corresponding question with an answer sentence selected from the news article. Recently, several deep neural networks have been proposed for sentence-level QA. For the best of our knowledge, none of them explicitly use keywords that appear simultaneously in questions and documents. In this paper we introduce the Attention-based Memory Network (Att-MemNN), a new iterative bi-directional attention memory network that predicts answer sentences. It exploits the co-occurrence of keywords among questions and documents as augment inputs of deep neural network and embeds documents and corresponding questions in different way, processing questions with word-level and contextual-level embedding while processing documents only with word-level embedding. Experimental results on the test set of NewsQA show that our model yields great improvement. We also use quantitative and qualitative analysis to show the results intuitively.

Keywords: Sentence-level question answering for news articles · Attention mechanism · Memory network · Deep learning

1 Introduction

Question answering (QA) for social media is a complex research problem in natural language processing because of the rapid growth of news articles and the diversity of text expressions in news article.

Our task is sentence-level QA which extracts an answer sentence from a document which is news article in QA for social media to answer a question based on the document. Many proposed works have focused on answer-sentence selection problems to leverage deep neural networks [1–4]. All of the models use datasets from the annual TREC evaluations [5] and WikiQA [6]. This kind of dataset provides a question and a set of candidate sentences and we should choose the best sentence from a candidate

X. Cheng et al. (Eds.): SMP 2017, CCIS 774, pp. 104–115, 2017.
https://doi.org/10.1007/978-981-10-6805-8_9

sentence set that can answer the question. Most recently, Trischler et al. (2016) present a challenging new large-scale dataset for machine comprehension: NewsQA [7], whose source material was chosen from CNN articles. Unlike TREC and WikiQA dataset, NewsQA provides a document and a question based on the document and we should answer the question with a sentence from the document. To explore NewsQA task, we propose a new iterative bi-directional attention neural network architecture. In our model, we explicitly use the keywords which appear simultaneously in the question and corresponding document. This idea is inspired from the fact that when human do the task of reading comprehension, some semantic words including verbs, nouns and all other words except prepositions and conjunctions in the question are critical clues to find the correct answer from the document, which we call them keywords. After identifying keywords in the question, human always find the same words in the document to answer the question. For example, given a question: *Who is Barcelona playing against?*, human always focus on "*Barcelona"* and "*play against"* and find the same words or words that represent the same meaning in the document. In a segment of corresponding document: *Barcelona has been in indifferent recent form and a 1-1 draw at Athletic Bilbao on Saturday. Barca will certainly want the key pair to be fit for next Sunday's El Clasico against Real.* The same word is the name of football club "*Barcelona*", and the "*Barca*" is the same meaning of "*Barcelona*", the "*against*" is the same meaning of "*playing against*" but have different spellings. Inspired by this way, we exploit the co-occurrence of words among questions and documents as augment input of our model, which has been reported to be one of the most important features for modeling question answering problem using a logistic regression model [8]. For the best of our knowledge, none of previous proposed deep neural networks takes the information of keywords as augment input of neural network. We index every sentence in the document with the keywords information into an index-vector and apply it to hidden representations of our neural attention model, which gets a noticeable improvement.

Our model can also be seen as a kind of Memory Network, generalizes the original Memory Network, MemN2N [9]. Both of the models have memory components to read from and write to, which can make iterative attention process. Our model offers following improvements to the benchmark model [9]. First, it explicitly uses keywords information and applies it to hidden representations in the memory network. Especially, due to the diversity of text expression in news articles, we use text normalization to transform documents and questions into a single canonical form, which helps to avoid missing the matching keywords in the document. Second, instead of uni-directional weight calculation in the baseline model, we use bi-directional attention mechanism in our model. We use a similarity matrix to calculate two different weights on both of document and question. The attention mechanism in our model is similar with it in MPCM model which only encodes a weighted document and an original question [10]. It is also similar with bi-directional attention flow in Bi-DAF network [11] whose target is to produce a set of question-aware feature vectors for each word in the document while our target is to produce a weighted document and a weighted question. Third, Sentence-level QA system always focus on every sentence in a document and temporal interactions between words in the document have less effect on our model. Therefore, we process questions with word-level and contextual-level embedding while process documents only with word-level embedding.

In this paper we introduce the Attention-based Memory Network (Att-MemNN), a new iterative bi-directional attention memory network architecture. It explicitly applies the keywords information to hidden representations in deep neural network and embeds documents and questions in different way. We perform experiments on the high-quality NewsQA dataset and our approach outperforms baseline methods by a significant amount. We also use quantitative and qualitative analysis to show the results intuitively.

2 Related Work

Recent years, many deep neural networks have been proposed for the QA task [12, 13], which have accelerated the progress of QA system. In this work we propose Att-MemNN model for sentence-level QA, and there are three main works which we are related to.

2.1 Question Answering System

Based on information retrieval, early QA systems were designed to return a segment of text from the corresponding reading document to answer a question which usually stuck in employing linguistic tools, feature engineering or other simple networks [8, 14]. However, without the use of deep natural networks all of the systems make a poor performance because of errors of many NLP tools and limitations of additional resources. Recently, many deep natural network models have been proposed for QA. From the way of identifying answers, most of the models can be roughly categorized into two classes: selecting the answer from a set of alternatives [15, 16] and extracting the answer from corresponding documents [17, 18]. In the former kind of method, we always extract candidate answers and train the model to rank the correct to the top of the list. The latter can be divided into sentence-level QA whose answer is a sentence from the corresponding document and span-level QA whose answer is a segment of text from the document. For span-level QA, Vinyals et al. (2015) use the Pointer Network to return a list of positions from the document as the final answer [19]. However, we cannot guarantee the selected positions to be consecutive. Xiong et al. (2016) introduce Dynamic Coattention Network (DCN) for question answering, which can recover from local maxima corresponding to incorrect answers [18].

2.2 Attention Based Models

Attention mechanisms are important in natural networks, which can significantly improve the performance of QA systems. There are many works have been done to show the effect of attention mechanisms [2, 20]. In attention based QA models, the representation of document is always built with attention from the representation of question which is uni-directional attention mechanism. Wang et al. (2016) use uni-directional attention mechanism in its model, adjusting each word-embedding vector in the document by multiplying a relevancy weight computed against the question [10]. Sukhbaatar et al. (2015) proposed a recurrent attention model with a

large external memory [9], which is also a kind of uni-directional attention mechanism. There are also some models represent questions with attention from the representation of documents [21]. To get a better performance, many QA systems start to use bi-directional attention mechanism in their model, which provide complimentary information to both of documents and the questions. Seo et al. (2017) proposed the Bi-Directional Attention Flow (BIDAF) network with the use of bi-directional attention flow mechanism which obtains the attentions and the attended vectors in both directions of document-to-question and question-to-document [11]. We use similar bi-directional attention mechanism in our model. However, the target of attention mechanism in BIDAF network is to produces a set of question-aware feature vectors for each word in the document while our target is to produce a weighted document and a weighted question.

2.3 Memory Networks

There are two difficulties in reading comprehension models: making multiple computational steps and representing long-term dependencies sequential. Many ways have been explored to exploit long-distance sequential information using RNNs or LSTM-based models which use the state of models to be memory [2, 22, 23]. However, the memory represented in that way is not stable over long timescales. Some works try to use global memory components in their models. Graves et al. (2014) proposed a Neural Turing Machine (NTM) model using a continuous memory representation [24]. However, the memory size in that model is small and the operation of sorting and recalling in NTM requires more complex models. Weston et al. (2014) proposed a Memory Network with a long-term memory component which enables multiple computational steps [25]. There are two deficiencies that the model requires supervision at each layer and is not easy to train via backpropagation algorithm. Sukhbaatar et al. (2015) proposed a continuous form of Memory Network, MemN2N which is trained end-to-end and requires less supervision [9]. Our model generalizes MemN2N model and offers some improvements to this benchmark model.

3 Model

In this section, we propose an Attention-based Memory Network (Att-MemNN) to estimate probability distribution P upon all of the sentences in the document to predict the answer sentence. Figure 1 shows the architecture of our model. Here the input of our model is a document and a corresponding question which are successively passed through embedding layer, multi-hops attention layer and output layer to get an answer sentence for the question as the output of our model. The keywords information module uses the document and the question to obtain an augment input for multi-hops attention layer and output layer.

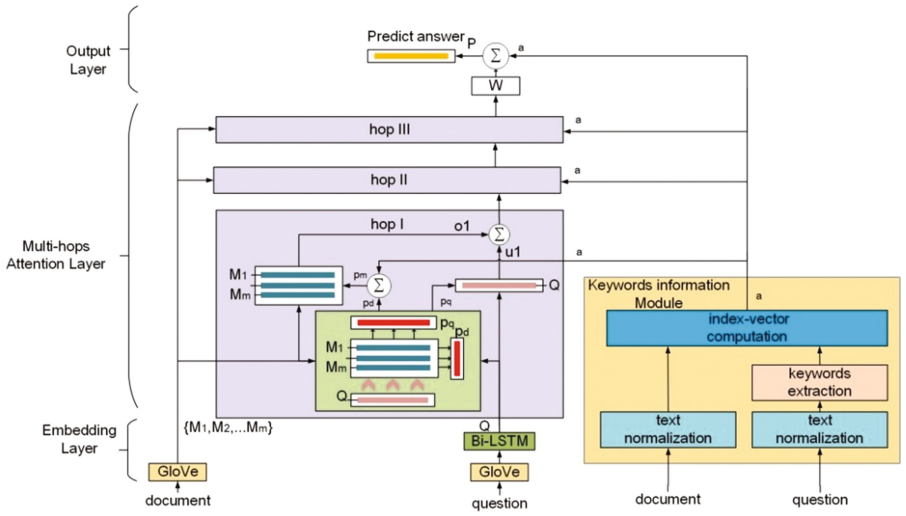

Fig. 1. Architecture of Attention Based Memory Network (Att-MemNN). Our model is stacked to multiple hops and is set to be 3 hops in this architecture.

3.1 Keywords Information Module

The target of this module is to represent keywords information. The "keywords" in this paper are semantic words that appear simultaneously in the document and corresponding question. To represent the keywords information, we propose an index-vector whose detail will be described in the following. The inputs of this module are raw texts of a document and corresponding question while the output is an index-vector which is sent to each hop of the attention layer and the output layer. The advantage of the module is to increase the weights of some sentences containing the same keywords with the question in the document. Though, for some examples, this way may results new noise, the adaptability of the model will reduce the impact of noise and experiments show that this way yields great improvement.

Firstly, we use text normalization to transform the document and question into a single canonical form [26], which makes the inputs of next step to be consistent texts to avoid missing the same keywords in document. Secondly, we extract keywords in the question. There are many ways to extract keywords such as simple statistic approach, linguistic approach, machine learning approach and hybrid approach [27]. In our model, the simple statistic approach is used to extract keywords in the question. To be simple, if words in the question are semantic words, they are chose to be keywords. Finally, determine which sentences in the document contain keywords. The event of word co-occurrence for each individual sentence is indicated by an (0, 1)-element of index-vector for the whole document. For example, given a question: *Where is Sonia Sotomayor?,* the keyword in this question is *Sonia Sotomayor*. Then, given a document: *Sonia Sotomayor goes to the bed room. Tom goes to the bathroom. Mary returns to the garden*, we index the first sentence containing *Sonia Sotomayor* with "1" and other sentences with "0" and obtain an index-vector [1, 0, 0]. In this module,

index-vector is represented by $a \in R^m$ consisting of 0 and 1 for each document where m is the maximum number of sentences of all the documents. If the number of sentences in a document is less than m, the index-vector will be padded with 0.

3.2 Embedding Layer

The target of this layer is to embed the document and question in different ways. As is shown in Fig. 1, the inputs of embedding layer are a question $q \in R^w$ and a document represented by a discrete set $s_1, s_2, \ldots, s_m$ where $s_i \in R^w$ represents the i-th sentence in the document and w is the maximum number of words of the sentences in the documents and questions. Each of the s_i, q contains w symbols coming from a dictionary which indexes every word in NewsQA dataset with a unique number. If the number of words in a sentence is less than w, s_i and q will be padded with symbol 0. The outputs of this layer are a question vector $Q \in R^e$ obtained from question q and memory vectors $\{M_i\}(M_i \in R^e)$ to represent the document obtained from a discrete set $\{s_i\}$.

When processing the question, we use word-level and contextual-level embedding. In the word-level embedding, we use pre-trained word vectors, GloVe [28], to represent every symbol in q with an e-dimensional continuous vector and obtain an intermediate matrix $q' \in R^{w \times e}$ for the question. We take the intermediate matrix q′ as input of contextual-level embedding. In contextual-level embedding, we place a Long Short-Term Memory Network (LSTM) in both directions to utilize contextual cues from surrounding words to refine the embedding of the words. We sum the outputs of the two LSTM together by which we get a matrix (of size $w \times$ e). Then, in order to convert the matrix into a vector, elements of the matrix were summed in column. In this way, we convert the question q into a question vector $Q \in R^e$. On account that sentence-level QA system always focus on every sentence in a document and temporal interactions between words in the document have less effect on our model, we doesn't process the document with contextual-level embedding. With the same way using in word-level embedding of the question, we embed a discrete set $\{s_i\}$ into memory vectors $\{M_i\}$.

3.3 Multi-hops Attention Layer

This is the core layer in our model and there is a memory component with shared read and write functions. In typical memory model, there are many memory input/output operations in the same way using in MemN2N [9]. To be simple, we write representation of the document into memory in embedding layer and read the memory in multi-hops attention layer many times. In this layer, the continuous memory representation for document and continuous representation for question are processed via multiple hops. In Fig. 1, it shows a model stacked to 3 hops and we simplify the graphical representation of the second and the third hop.

In each hop, we use a bi-directional attention mechanism on both of the question and the document stored in the memory. To calculate the memory weight p_m on the document and question weight p_q on the question, we firstly calculate a similarity matrix $S \in R^{m \times e}$ by taking the inner product of Q and $\{M_i\}$. S_{ij} is a numerical value

which indicates the similarity between i-th element in question and j-th element of i-th sentence in the document:

$$S_{ij} = Q_i M_{ij}. \tag{1}$$

By similarity matrix, we can easily get a document weight $p_d \in R^m$ which indicates which sentences in the document are more relevant to the question. Because the i-th line of the similarity matrix represents the similarity between each element of the question and i-th sentence in the document, we sum elements of the similarity matrix in row to get a document weight $p_d \in R^m$ for every sentence in the document:

$$p_d = Softmax(\sum\nolimits_j S_{ij}). \tag{2}$$

The index-vector $a \in R^m$ obtained by keywords information module is an augment input for this layer. It models the event of keywords co-occurrence and can also measure which sentences in the document are more relevant to the question. Although p_d is somewhat an indication of the similar relation between question and the sentences in document which is usually adopted by previous attention mechanism, the index-vector will enhance the similar relation as an explicit prior knowledge. Therefore, we sum index-vector to document weight p_d by which we increase the weights of sentences in document containing keywords of corresponding question. Then, the document weight p_d was updated to the memory weight $p_m \in R^m$:

$$p_m = Softmax(p_d + a). \tag{3}$$

By using attention mechanism on every sentence in the memory with the memory weight p_m, we obtain a response vector $o \in R^e$ from the memory vectors $\{M_i\}$:

$$o = \sum\nolimits_i p_{m_i} M_i. \tag{4}$$

In a similar way, we add elements of the similarity matrix in column to get a question weight $p_q \in R^e$ for the question vector Q and weight every element in the question to obtain an internal state $u \in R^e$ from question vector Q:

$$p_q = Softmax(\sum\nolimits_i S_{ij}). \tag{5}$$

$$u = p_q \circ Q. \tag{6}$$

where o in the formula represents Hadamard product.

The output of this hop is $(u \cdot H + o)$ where H is a trainable matrix of size $e \times e$. This output is inputted to the next hop as the question vector Q of the next hop. Every hop in our model has the same architecture and $\{M_i\}$ is obtained by memory output operations in each hop. The output of the modeling layer is the response vector o and the internal state u of the last hop.

3.4 Output Layer

The target of the output layer is to estimate probability distribution on all of the sentences in the document and predict the answer sentence. The sum of the output vector o and internal state u is then passed through a final weight matrix $W \in R^{e \times m}$ and a softmax to get an intermediate probability $P_i \in R^m$:

$$P_i = Softmax(W(o + u)). \tag{7}$$

The index-vector containing keywords information is utilized in the probability distribution, which can increase the weights of sentences containing keywords of corresponding question and eliminate interference from other sentences. The index-vector a is added to the intermediate probability to produce final predicted probability P:

$$P = Softmax(P_i + a). \tag{8}$$

The predicted probability P is used to predict the answer sentence.

During training, the training loss (to be minimized) is defined as the standard cross-entropy loss between predicted probability P and the true probability P'. The matrix W and H are jointly learned when the training is performed using stochastic gradient descent. During testing, the sentence with the maximum probability is chosen, computed by the predicted probability P.

4 Experiment

We conducted our experiments on the NewsQA dataset to evaluate the performance of our model.

4.1 Dataset

NewsQA is a crowd-sourced machine comprehension dataset on a large set of CNN articles. The number of average words per article is 616 from which we can see the article in NewsQA is large volumes of text. To evaluate our model, we use accuracy, can also be seen as "Exact match (EM)", which calculate the ratio of questions that are answered correctly. We also use F1-Measure to evaluate models which is calculated by precision and recall. In our experiment, searching the wide space of possible configurations is quite costly because of the size of the dataset. To alleviate this, we randomly select 3221 question-answer pairs to train the model and 546 question-answer pairs evaluated the performance of model. The NewsQA dataset is for span-level QA systems, we extract sentences containing answer spans to be the answer of corresponding questions. In batch tests, we randomly divide the test set into three parts. Each part contains 182 question-answer pairs. In addition similar results are obtained from all parts. The maximum number of sentences of all the documents (represent by m in our model) is 152 and the maximum number of words of the sentence in all the documents and questions (represent by w in our model) is 155.

4.2 Model Setup

In embedding layer, we use 50-dimensional vector to represent each word in documents and questions. We use the Adam optimizer, with an initial learning rate of 0.01 and an epsilon value of 1e−8. No momentum or weight decay was used. We use a batch size of 32 in all training, and the maximum gradient norm is 40. The gradient in training is clip to this norm. A dropout rate of 0.26 is used for the model. Since the number of sentences and the number of words are constrained into fixed size, we use some null symbol to pad them. The training process takes roughly 480 min on a single NVIDIA GPU.

Because of the random initialization, the result of every training process is different. To remedy this, we repeated the training for five times and picked the best result as the final result.

4.3 Results and Analysis

The results of our model and competing approaches are shown in Table 1. We evaluated our model with accuracy, can also be seen as "Exact match (EM)"and F1-Measure. In Table 1, the inverse sentence frequency (ISF) model is proposed by [11], which is a technique that resembles inverse document frequency (idf). The MemN2 N model is proposed by [13] used for bAbI task and we make minor modifications to apply it for sentence-level QA task. As we can see, the accuracy of our model is 61.3% exceeding MemN2N by 29.2% and ISF by 25.9%. Our model yields improved results.

Table 1. The performance of our model Att-MemNN and competing approach including ISF [7] and MemN2N [9]. Memory module of our model is set to 3 hops.

Model	Accuracy (EM)	F1
ISF [11]	35.4%	
MemN2N [9]	32.1%	38.3%
Att-MemNN (ours)	61.3%	69.1%

Model Ablation. We also propose an ablation subtask to evaluate the effectiveness of various improvements in our Att-MemNN model. In introduction of this paper, we have proposed three improvements of our model. We remove one improvement at a time to perform the experiment. When removing different embedding, we use a trainable embedding matrix to embed documents and questions which is used in benchmark model, MemN2N [9]. Table 2 shows the results of all ablation models and our full model on NewsQA. We can see that each of the components have effect on the model. Removing keywords information module reduces the accuracy by 22.7% and F1 by 22.5%. Changing bi-attention mechanism into uni-attention mechanism reduces the accuracy by 5.6% and F1 by 6.6%. Removing different embedding reduces the accuracy by 5.1% and F1 by 5.6%. Among all the components, removing the keywords information module decreases the performance significantly. It indicates that keywords information has the biggest promotion for our model among the three improvements.

Table 2. Test accuracy on ablation experiment. Memory module of our model is set to 3 hops.

Model	Accuracy	F1
Att-MemNN	61.3%	69.1%
No keywords information	38.6%	46.6%
No bi-attention	55.7%	62.5%
No different embeddings	56.2%	63.6%

Table 3. Effectiveness of multiply hops memory module on NewsQA.

Model	Accuracy	F1
1 hop	47.5%	55.7%
2 hops	55.6%	61.6%
3 hops	61.3%	69.1%

Analysis of Multiply Hops Memory Module. We put quantitative and qualitative analysis on multiply hops memory module which is an important part of our model. Table 3 shows the effectiveness of multiply hops memory module on NewsQA. Memory module of our model is set to 1 hop, 2 hops and 3 hops. We note that for NewsQA, multiply hops memory module which enable iterative attention are crucial to achieving high performance. Figure 2 shows the attention weights on every sentence in a document for 1-hop model and 3-hops model. This example demonstrates that the multi-hops memory module allows the model sharply focus on relevant sentences.

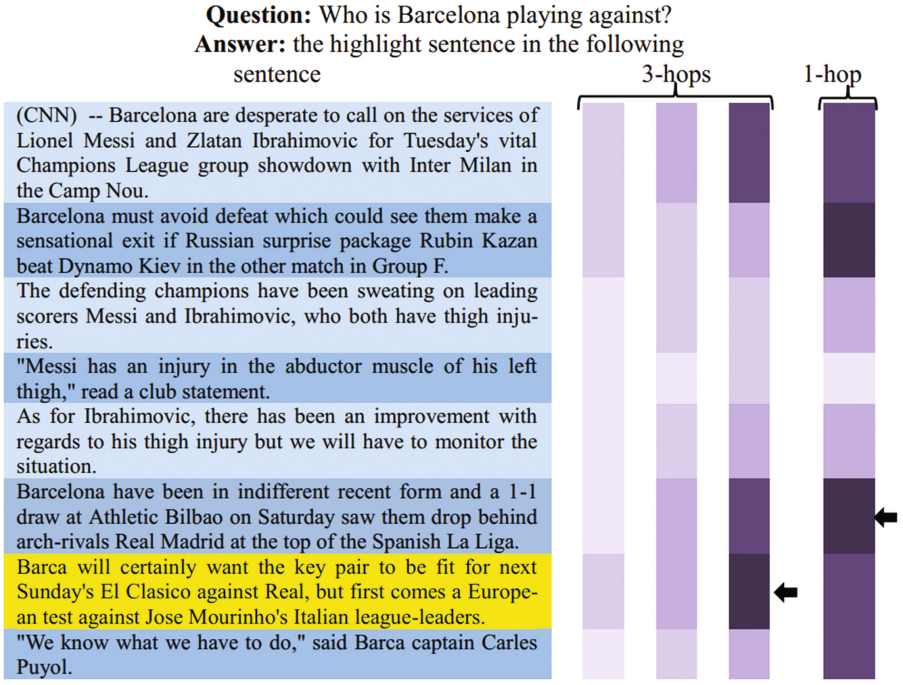

Fig. 2. Attention weights on every sentence in a document for 1-hop model and 3-hops model. In 3-hops model, it shows attention weights of the first hop, second hop and third hop from left to right. Color deepness in the picture means different weight. The sentence indicated by the arrow is the predicted answer sentence of the model. (Color figure online)

Table 4. Analysis on keywords information module.

Question	Answer
Q: What is going live on **Tuesday**?	A: Comcast rolled out a Web-based on-demand television and movie service on **Tuesday**
Q: Who is **Sonia Sotomayor**?	A: Judge **Sonia Sotomayor**, center, meets with staffers from the White House Counsel's Office
Q: What was the **space station crew forced** to take **shelter** from?	A: Last week, a piece of debris **forced** the **space station's** current **crew**
Q: What does **Gary go use** for **musical accompaniment**?	A: **Gary go uses** iPhone apps to help him compose new material – and provide instant **accompaniment**

Analysis of Keywords Information Module. To show the effect of keywords information, we output results of keywords extraction module in our model. Table 4 shows some examples on keywords information. We randomly select several question-answer pairs from test set of NewsQA and highlight keywords in questions extracted by keywords extraction module and the corresponding keywords in answer sentence. The result of Table 4 and statistics suggest that 70.5% correct sentences contain keywords in questions, which show that keywords information is useful for sentence-level QA system.

5 Conclusion and Future Work

In this paper, we proposed Att-MemNN, a new bi-directional attention memory network that predicts the answer sentence from a news article to answer a corresponding question. The model explicitly uses the information of keywords that appear simultaneously in questions and documents and represents documents and questions in different way. Experimental results on the test set of NewsQA show that our model yields improved results. The ablation analyses show the importance of each improvement in our model. In the future, we can add a module to the model by which we can obtain the exact answer from the sentence chosen from our model.

Acknowledgments. This work was supported by NSF Projects 61602048, 61302077.

References

1. Tymoshenko, K., Bonadiman, D., Moschitti, A.: Convolutional neural networks vs. convolution kernels: feature engineering for answer sentence reranking. In: Proceedings of NAACL-HLT, pp. 1268–1278 (2016)
2. Wang, B., Liu, K., Zhao, J.: Inner attention based recurrent neural networks for answer selection. In: The Annual Meeting of the Association for Computational Linguistics (2016)
3. Zheng, Z.: AnswerBus question answering system. In: Proceedings of the Second International Conference on Human Language Technology Research, pp. 399–404. Morgan Kaufmann Publishers Inc., March 2002

4. Tahri, A., Tibermacine, O.: DBPedia based factoid question answering system. Int. J. Web Semant. Technol. **4**(3), 23 (2013)
5. Wang, M., Smith, N.A., Mitamura, T.: What is the Jeopardy Model? A quasi-synchronous grammar for QA. In: EMNLP-CoNLL, vol. 7, pp. 22–32, June 2007
6. Yang, Y., Yih, W.T., Meek, C.: WikiQA: a challenge dataset for open-domain question answering. In: EMNLP, pp. 2013–2018, September 2015
7. Trischler, A., Wang, T., Yuan, X., Harris, J., Sordoni, A., Bachman, P., Suleman, K.: NewsQA: a machine comprehension dataset. arXiv preprint arXiv:1611.09830 (2016)
8. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250 (2016)
9. Sukhbaatar, S., Weston, J., Fergus, R.: End-to-end memory networks. In: Advances in Neural Information Processing Systems, pp. 2440–2448 (2015)
10. Wang, Z., Mi, H., Hamza, W., Florian, R.: Multi-perspective context matching for machine comprehension. arXiv preprint arXiv:1612.04211 (2016)
11. Seo, M., Kembhavi, A., Farhadi, A., Hajishirzi, H.: Bidirectional attention flow for machine comprehension. arXiv preprint arXiv:1611.01603 (2016)
12. Wang, W.H., Yang, N., Wei, F.R., et al.: Gated self-matching networks for reading comprehension and question answering. In: ACL (2017)
13. Hu, M., Peng, Y., Qiu, X.: Mnemonic reader for machine comprehension. arXiv preprint arXiv:1705.02798 (2017)
14. Richardson, M., Burges, C.J., Renshaw, E.: MCTest: a challenge dataset for the open-domain machine comprehension of text. In: EMNLP, vol. 3, p. 4, May 2013
15. Yu, Y., Zhang, W., Hasan, K., Yu, M., Xiang, B., Zhou, B.: End-to-end answer chunk extraction and ranking for reading comprehension. arXiv preprint arXiv:1610.09996 (2016)
16. Lee, K., Kwiatkowski, T., Parikh, A., Das, D.: Learning recurrent span representations for extractive question answering. arXiv preprint arXiv:1611.01436 (2016)
17. Wang, S., Jiang, J.: Machine comprehension using match-LSTM and answer pointer. arXiv preprint arXiv:1608.07905 (2016)
18. Xiong, C., Zhong, V., Socher, R.: Dynamic coattention networks for question answering. arXiv preprint arXiv:1611.01604 (2016)
19. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks. In: Advances in Neural Information Processing Systems, pp. 2692–2700 (2015)
20. Zhang, W.M., Zhang, C.H., Liu, P., Zhan, Z.Q., Qiu, X.F.: Two-step joint attention network for visual question answering. In: BIGCOM (2017)
21. Sordoni, A., Bachman, P., Trischler, A., Bengio, Y.: Iterative alternating neural attention for machine reading. arXiv preprint arXiv:1606.02245 (2016)
22. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
23. Graves, A.: Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850 (2013)
24. Graves, A., Wayne, G., Danihelka, I.: Neural turing machines. arXiv preprint arXiv:1410.5401 (2014)
25. Weston, J., Chopra, S., Bordes, A.: Memory networks. Eprint Arxiv (2014)
26. Clark, E., Araki, K.: Text normalization in social media: progress, problems and applications for a pre-processing system of casual English. Procedia Soc. Behav. Sci. **27**, 2–11 (2011)
27. Bharti, S.K., Babu, K.S.: Automatic keyword extraction for text summarization: a survey. arXiv preprint arXiv:1704.03242 (2017)
28. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP 2014, pp. 1532–1543 (2014)