Memory-Based Matching Models for Multi-turn Response Selection in Retrieval-Based Chatbots

Xingwu Lu[1], Man Lan[1,2](✉), and Yuanbin Wu[1,2](✉)

[1] School of Computer Science and Software Engineering, East China Normal University, Shanghai 200062, People's Republic of China
51174506023@stu.ecnu.edu.cn, {mlan,ybwu}@cs.ecnu.edu.cn

[2] Shanghai Key Laboratory of Multidimensional Information Processing, Shanghai, China

Abstract. This paper describes the system we submitted to Task 5 in NLPCC 2018, i.e., Multi-Turn Dialogue System in Open-Domain. This work focuses on the second subtask: Retrieval Dialogue System. Given conversation sessions and 10 candidates for each dialogue session, this task is to select the most appropriate response from candidates. We design a memory-based matching network integrating sequential matching network and several NLP features together to address this task. Our system finally achieves the precision of 62.61% on test set of NLPCC 2018 subtask 2 and officially released results show that our system ranks 1st among all the participants.

Keywords: Multi-turn conversation · Response selection
Neural networks

1 Introduction

Recently, more and more attention is paying to building open domain chatbots that can naturally converse with humans on vary topics. Existing work on building chatbots includes generation-based methods [1–3] and retrieval-based methods [4–7]. Compared to generation-based chatbots, retrieval-based chatbots enjoy the advantages of informative and fluent responses, because they select a proper response for the current conversation from a repository.

Different from the single-turn conversation, multi-turn conversation needs to consider not only the matching between the response and the input query but also matching between the response and context in previous turns. The challenges of the task are how to identify important information in previous utterances and properly model the utterances relationships to ensure the consistency of conversation.

There have been many attempts to address these challenges where the state-of-the-art methods include dual LSTM [4], Multi-View LSTM [6], Sequential

M. Zhang et al. (Eds.): NLPCC 2018, LNAI 11108, pp. 269–278, 2018.
https://doi.org/10.1007/978-3-319-99495-6_23

Matching Network (SMN) [7] and so on. Among them, SMN improves the leveraging of contextual information by matching a response with each utterance in the context on multiple levels of granularity with a convolutional neural network, and then accumulates the matching vectors into a chronological order through a recurrent neural network to model sequential relationships among utterances. Although SMN model has achieved remarkable results, there are still problems of inconsistency between response and context. On the one hand, the context of the dialogue may sometimes be complex. For example, some utterances are interrelated and some are even reversed. On the other hand, important context cues require global information to be captured.

In this work, based on SMN, we develop a novel way of applying multiple-attention mechanism, which is proven to be effective in multiple tasks such as sentiment analysis [8,9], dependency parsing [10] and coherence modeling [11]. Different from the SMN that only considers the sequential relationships of the context, our method also synthesizes important features in complex context. Besides, considering the effectiveness of NLP features in some retrieval tasks [12], we also design several effective NLP features. Specifically, our framework first adopts matching vectors to produce the memory. After that, we pay multiple attentions on the memory and nonlinearly combine the attention results with a recurrent network, i.e. Long Short-term Memory (LSTM) [13] networks. Finally, we combine the output of the LSTM network with the output of SMN and NLP features to calculate the final matching score. Our system finally achieves the precision of 62.61% on the test set of NLPCC 2018 Task 5[1] and ranks 1st among all the participants.

The rest of this paper is structured as follows: we describe the system architecture and detailed modules in Sect. 2, and present the experimental results in Sect. 3. Finally, Sect. 4 presents our conclusion and future work.

2 The Approach

2.1 Model Overview

The architecture of our model is shown in Fig. 1, which consists of three main modules, i.e., SMN, MBMN and NLP. The left red wire frame is sequential matching network (SMN) module, which is based on [7]. It is designed to identify important information in previous utterances and model the sequential relationships in context. Considering that SMN may not capture implied and complex contextual features, we design the memory-based matching network (MBMN) module, i.e., the middle green wire frame. As shown in the right blue wire frame, we also design several effective NLP features in NLP features module since it is proved to be effective in some retrieval tasks. Finally, we concatenate the outputs of these three modules in the matching prediction layer to calculate the final matching score. Next we give detailed description.

[1] http://tcci.ccf.org.cn/conference/2018/dldoc/taskgline05.pdf

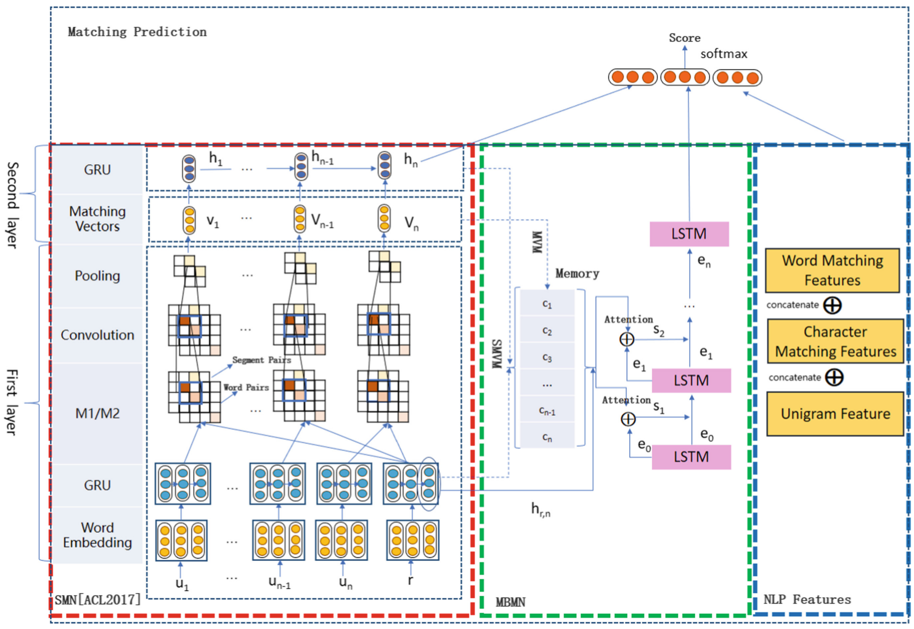

Fig. 1. System architecture of our approach. The dotted lines on MBMN module indicate the memory building is alternative.

2.2 Sequential Matching Network (SMN)

We follow [7] and design the SMN module. The source code[2] of SMN is released by [7]. This module has two advantages: (1) identify and extract semantic structures that are useful for response selection in each utterance and (2) model chronological relationships of the context. As shown on the left of Fig. 1, this module is divided into two layers: utterance-response matching layer (first layer) and matching accumulation layer (second layer). The two layers reflect the above two advantages respectively and their implementations are shown as follows:

- *Utterance-Response Matching Layer:* This layer matches a response candidate with each utterance in the context on a word level and a segment level, and the important matching information from the two levels is distilled by convolution, pooling and then encoded into a matching vector.
- *Matching Accumulation Layer:* We feed the matching vectors into the matching accumulation layer where they are accumulated in the hidden states of a recurrent neural network with gated recurrent units (GRU) [14] following the chronological order of the utterances in the context.

[2] https://github.com/MarkWuNLP/MultiTurnResponseSelection.

2.3 Memory-Based Matching Network (MBMN)

The SMN model only considers semantic structures and chronological relationships in utterances, ignoring the important features in complex context. Herein, we design MBMN module to distill cue information that should be captured by global context information and some important contextual information that have long-distance dependence on the query. These cue and important information are captured and retained by memory.

2.3.1 Memory Building

In order to explore the effectiveness of memory, we use two different ways to build memory: *matching vectors memory* (MVM) and *sequence matching vectors memory* (SMVM). We define the representation of memory as $[c_1, \ldots, c_n]$ and their implementation are shown as follows:

- *Matching Vectors Memory (MVM):* Suppose that matching vectors $[v_1, \ldots, v_n]$ is the output of the first layer in SMN module, we directly use the matching vectors as memory vectors, i.e., $[c_1, \ldots, c_n] = [v_1, \ldots, v_n]$.
- *Sequence Matching Vectors Memory (SMVM):* MVM simply uses the matching vectors as memory, which ignores the sequential features in the context. Considering the sequential features dominate in dialogue utterances, we use the hidden states of final GRU in second layer and utterance GRU in first layer of SMN module to build the memory. Then, $[c_1, \ldots, c_n]$ is defined as

$$c_i = tanh(W_{1,1} h_{u_i,n_u} + W_{1,2} h_i + b_i) \tag{1}$$

  where $W_{1,1} \in \mathbb{R}^{q \times p}$, $W_{1,2} \in \mathbb{R}^{q \times q}$ and $b_1 \in \mathbb{R}^q$ are parameters, p is the hidden size of utterance GRU, q is the hidden size of final GRU, h_i and h_{u_i,n_u} are the i-th hidden states of final GRU and the final hidden state of the i-th utterance GRU respectively.

2.3.2 Multiple Attentions on Memory

To accurately select the candidate response, it is essential to: (1) correctly distill the related context information from its utterance-response matching memory; and (2) appropriately manufacture such information as the input of the matching prediction. We employ multiple attentions to fulfil the first aspect, and a recurrent network for the second aspect which nonlinearly combines the attention results with LSTMs.

Particularly, we employ a LSTM to update the episode e (i.e., hidden state of LSTM) after each attention. Let e_{t-1} denote the episode at the previous time and s_t is the current information attended from the memory C, and the process of updating e_t is as follows:

$$i_t = \sigma(W_i s_t + U_i e_{t-1}) \tag{2}$$

$$f_t = \sigma(W_f s_t + U_f e_{t-1}) \tag{3}$$

$$o_t = \sigma(W_o s_t + U_o e_{t-1}) \tag{4}$$

$$g = tanh(W_c s_t + U_c e_{t-1}) \tag{5}$$

$$C_t = f \odot C_{t-1} + i \odot g \tag{6}$$

$$h_t = o_t \odot tanh(C_t) \tag{7}$$

where $W_i, W_f, W_o, W_c \in \mathbb{R}^{h\times c}$, $U_i, U_f, U_o, U_c \in \mathbb{R}^{h\times h}$ are parameters, h is the hidden size of LSTM and c is the size of memory vector c_i, a zero vector is denoted as e_0.

For calculating the attended information s_t at time t, the input of an attention layer includes the memory slices $c_i(1 \leq i \leq N)$, N is the number of utterances, the previous episode e_{t-1} and $h_{r,n}$, which is the final hidden state of the response GRU in the first layer of SMN module. We first calculate the attention score of each memory slice c_i as follows:

$$g_i^t = v^T tanh(W_c c_i + W_e e_{t-1} + W_r h_{r,n} + b_{attn}) \tag{8}$$

where W_c, W_e, W_r and b_{attn} are parameters.

Then we calculate the normalized attention score of each memory slice as:

$$\alpha_{t_i} = \frac{exp(g_i^t)}{\sum_{j=1}^{T} exp(g_j^t)} \tag{9}$$

Finally, the inputs to a LSTM at time t are the episode e_{t-1} at time $t-1$ and the content s_t, which is read from the memory as:

$$s_t = \sum_{i=1}^{N} \alpha_{t_i} c_i \tag{10}$$

2.4 NLP Features

This task provides 10 candidate responses corresponding to the context in test dataset and participants are required to rerank the candidates and return the top one as a proper response to the context. We connect all utterances as a post and measure the matching level of the post and its candidate response. We design several traditional NLP features to capture the relevance between the post context and their candidate response. The details of these features are shown as follows:

- *Word Matching Feature:* Word is the basic unit of sentence and the matching of word level benefits the matching of sentence level. Given the post and response as A and B, we record the matching information using the following ten measure functions: $|A|$, $|B|$, $|A\cap B|$, $|A\cap B|/|A|$, $|A\cap B|/|B|$, $|A-B|/|A|$, $|A-B|/|B|$, $|A\cap B|/|A\cup B|$, $|A\cup B| - |A\cap B|/|A\cup B|$, $||A| - |B||$, where $|A|$ stands for the number of non-repeated words in A, $|A-B|$ means the number of non-repeated words found in A but not in B, $|A\cap B|$ stands for the set size of non-repeated words found in both A and B, and $|A\cup B|$ means the set size of shared words found in either A or B.

- *Character Matching Feature:* Similar to word matching, all sentences are treated as the set of single-character representations, then we use above ten measure functions to represent character matching.
- *Unigram Feature:* We extract unigram to represent each sentence, and each vector stores the corresponding TF-IDF of the words in the sentence. We adopt kernel functions to calculate sentence pair matching score. Here we use two types of kernel functions: linear and non-linear. The liner functions contain *Cosine*, *Chebyshev*, *Manhattan*, and *Euclidean*. And the non-liner functions contain *Polynomial*, *Sigmoid* and *Laplacian*.

2.5 Matching Prediction

The representations of above three modules described in Sects. 2.2, 2.3 and 2.4 are concatenated (denoted as $[p_1, p_2, p_3]$) to calculate the final matching score $g(u, r)$. We define u_i represents a conversation context, r_i represents a response candidate and $y_i \in \{0, 1\}$ denotes label. Then we use *softmax* to obtain the final matching score $g(u, r)$ as follows:

$$g(u, r) = softmax(W_2[p_1, p_2, p_3] + b_2), \tag{11}$$

where W_2 and b_2 are parameters.

We learn $g(u, r)$ by minimizing *cross entropy* with dataset D. Let Θ denotes the parameters, then the objective function $L(D, \Theta)$ of learning is formulated as:

$$-\sum_{i=1}^{|D|}[y_i log(g(u_i, r_i)) + (1 - y_i)(1 - log(g(u_i, r_i)))] \tag{12}$$

2.6 Parameter Learning

All models are implemented using Tensorflow. We train word embeddings on the training data using word2vec [15] and the dimensionality of word vectors is set as 200. As previous works did [7], we set the hidden size of utterance GRU and response GRU as 200, window size of convolution and pooling as (3, 3), the number of feature maps as 8 and the dimensionality of matching vectors as 50. Different from [7], we tune the hidden size of final GRU in second layer of SMN module in [50, 100, 200, 300] and choose 200 finally. The LSTM in the MBMN module uses a hidden size of 200. We try the number of attention cycles in [1, 3, 5, 7, 9] and set 5 finally. The parameters are updated by stochastic gradient descent with Adam algorithm [16] and the initial learning rate is 0.001. We employ early stop as a regularization strategy. Models are trained in mini-batches with a batch size of 256. Hyperparameters are chosen using the validation set.

3 Experiments

3.1 Datasets

Specifically, this task provides $5,000,000$ conversation sessions containing context, query and reply as the training set and extra $10,000$ conversation sessions only contain context and query as the test set. Participants are required to select a appropriate reply from 10 candidates corresponding to the sessions in test set. Examples of the datasets are shown in Table 1.

Table 1. Data format of multi-turn response selection examples.

Training Set	**Test Context**	**Test candidates**
Context: 谢谢你所做的一切 Context: 你开心就好 Context: 开心 Context: 嗯因为你的心里只有学习 Query: 某某某，还有你 Reply: 这个某某某用的好	Context: 你能看下去么 Context: 昨晚已经看完了 Context: 我看睡着了 Context: 爱情啊菇凉 Query: 太静谧了这电影	(1) 对对对，超级喜欢 (2) 到底是为什么 (3) 我也是后学的 (4) 静静的去感受 (5) 关注啦 (6) 呵呵，有意思 (7) 我建议以后单眼皮一对你发嗲，你就跟她对发，哈哈 (8) 不要得瑟你 (9) 感觉好冷 (10) 有钱淫

We randomly split the data into 4,960,000/40,000 for training/validation. For each dialogue in training and validation set, we take the reply as a positive response for the previous turns as a context and randomly sample another response from the 5 million data as a negative response. The ratio of the positive and the negative is 1:1 in training set, and 1:9 in validation set. The word-segmentation is obtained with jieba[3]. We set the maximum context length (i.e., number of utterances) as 10. We pad zeros if the number of utterances in a context is less than 10, otherwise we keep the last 10 utterances. Table 2 gives the statistics of the training set, validation set and test set.

Table 2. Statistics of the training set, validation set and test set.

	Train	Val	Test
# context-response pairs	9.92M	400K	100K
# candidates per context	2	10	10
# positive candidates per context	1	1	1
Max. # turns per context	86	50	34
Avg. # turns per context	3.10	3.07	3.10
Max. # words per utterance	135	93	94
Avg. # words per utterance	5.97	6.22	6.28

To evaluate the performance, given 10 candidates, we calculate precision at top 1.

[3] https://github.com/fxsjy/jieba.

3.2 Experiments on Training Data

In order to explore the effectiveness of each module, we perform a series of experiments. Table 3 lists the comparison of different modules on training set. We observe the following findings:

(1) The MBMN(SMVM) performs the best among all single models. The performance of MBMN(MVM) is lower than SMN. The possible reason may be that SMN captures the sequential relationship of utterances in the context and sequential relationship plays a dominant role in this dialogues context.
(2) The memory-based matching modules are quite effective. The combined model MBMN(MVM)+SMN performs better than any single model. It indicates that the memory-based matching module is able to distill the cue information captured by global information in complex context rather than sequential context alone.
(3) The performance of model MBMN(MVM)+SMN is comparable to that of MBMN(SMVM)+SMN. It shows that the MBMN(SMVM) model itself has taken advantage of sequential features and its combination with SMN may not significantly improve the performance.
(4) The combination of three modules, i.e., MBMN(MVM)+SMN+NLP, achieves the best performance, which proves the effectiveness of our designed NLP features.

Therefore, the system configuration for our final submission is the combined model of MBMN(MVM)+SMN+NLP.

Table 3. Performance of different modules on validation set. (MVM) means the model based on matching vectors memory, (SMVM) means the model based on sequence matching vectors memory and "+" means module combination.

	Model	Precision (%)
Single model	NLP features	39.67
	SMN [ACL2017]	61.76
	MBMN(MVM)	60.03
	MBMN(SMVM)	**61.97**
Combined model	MBMN(MVM)+SMN	62.11
	MBMN(SMVM)+SMN	62.08
	MBMN(MVM)+SMN+NLP	**62.26**
	MBMN(SMVM)+SMN+NLP	62.16

3.3 Results on Test Data

Table 4 shows the results of our system and the top-ranked systems provided by organizers for this Retrieval Dialogue System task. Our system finally achieves the precision of 62.61% on the test set and ranks 1st among all the participants. This result validates the effectiveness of our model.

Table 4. Performance of our system and the top-ranked systems in terms of precision (%). The numbers in the brackets are the official rankings.

Team ID	Precision (%)
ECNU	62.61 (1)
wyl_buaa	59.03 (2)
YiwiseDS	26.68 (3)

4 Conclusion

In this paper, we design three modules of sequential matching network, memory-based matching network and NLP features to perform multi-turn response selection in retrieval dialogue system. The system performance ranks 1st among all the participants. In future work, we consider to design more effective memory to incorporate the location and inner semantic information of context in dialogues.

Acknowledgements. The authors would like to thank the task organizers for their efforts, which makes this event interesting. And the authors would like to thank all reviewers for their helpful suggestions and comments, which improve the final version of this work. This work is supported by the Science and Technology Commission of Shanghai Municipality Grant (No. 15ZR1410700) and the open project of Shanghai Key Laboratory of Trustworthy Computing (No. 07dz22304201604).

References

1. Shang, L., Lu, Z., Li, H.: Neural responding machine for short-text conversation. arXiv preprint arXiv:1503.02364 (2015)
2. Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., Nie, J.-Y., Gao, J., Dolan, B.: A neural network approach to context-sensitive generation of conversational responses. arXiv preprint arXiv:1506.06714 (2015)
3. Serban, I.V., Sordoni, A., Bengio, Y., Courville, A.C., Pineau, J.: Building end-to-end dialogue systems using generative hierarchical neural network models. In: AAAI, vol. 16, pp. 3776–3784 (2016)
4. Lowe, R., Pow, N., Serban, I., Pineau, J.: The Ubuntu dialogue corpus: a large dataset for research in unstructured multi-turn dialogue systems. Computer Science (2015)
5. Yan, R., Song, Y., Wu, H.: Learning to respond with deep neural networks for retrieval-based human-computer conversation system. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 55–64. ACM (2016)
6. Zhou, X., Dong, D., Wu, H., Zhao, S., Yu, D., Tian, H., Liu, X., Yan, R.: Multi-view response selection for human-computer conversation. In: Conference on Empirical Methods in Natural Language Processing, pp. 372–381 (2016)
7. Wu, Y., Wu, W., Xing, C., Zhou, M., Li, Z.: Sequential matching network: a new architecture for multi-turn response selection in retrieval-based chatbots (2017)
8. Tang, D., Qin, B., Liu, T.: Aspect level sentiment classification with deep memory network, pp. 214–224 (2016)

9. Chen, P., Sun, Z., Bing, L., Yang, W.: Recurrent attention network on memory for aspect sentiment analysis. In: Conference on Empirical Methods in Natural Language Processing, pp. 452–461 (2017)
10. Zhang, Z., Liu, S., Li, M., Zhou, M., Chen, E.: Stack-based multi-layer attention for transition-based dependency parsing. In: Conference on Empirical Methods in Natural Language Processing, pp. 1677–1682 (2017)
11. Logeswaran, L., Lee, H., Radev, D.: Sentence ordering and coherence modeling using recurrent neural networks (2017)
12. Tay, Y., Phan, M.C., Tuan, L.A., Hui, S.C.: Learning to rank question answer pairs with holographic dual LSTM architecture (2017). https://doi.org/10.1145/3077136.3080790. arXiv arXiv:1707 (2017)
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
14. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
15. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
16. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)