Response Selection of Multi-turn Conversation with Deep Neural Networks

Yunli Wang[1], Zhao Yan[2], Zhoujun Li[1(✉)], and Wenhan Chao[1]

[1] Beihang University, Beijing, China
{wangyunli,lizj,chaowenhan}@buaa.edu.cn
[2] Tencent, Beijing, China
zhaoyan@tencent.com

Abstract. This paper describes our method for sub-task 2 of Task 5: multi-turn conversation retrieval, in NLPCC2018. Given a context and some candidate responses, the task is to choose the most reasonable response for the context. It can be regarded as a matching problem. To address this task, we propose a deep neural model named RCMN which focus on modeling relevance consistency of conversations. In addition, we adopt one existing deep learning model which is advanced for multi-turn response selection. And we propose an ensemble strategy for the two models. Experiments show that RCMN has good performance, and ensemble of two models makes good improvement. The official results show that our solution takes 2nd place. We open the source of our code on GitHub, so that other researchers can reproduce easily.

Keywords: Multi-turn conversation · Response selection
Relevance consistency

1 Introduction

The task 5 of NLPCC 2018 focus on how to utilize context to conduct multi-turn human-computer conversations. It contains two sub-tasks: response generation and response retrieval. We signed up the response retrieval task, which is to select the most reasonable response for context from some given candidates. The data set is real multi-turn human-to-human conversations in Chinese, and it is in open domain.

The retrieval task can be regarded as a matching problem, which is to give a matching score between context and response. The challenges of this task are how to make full use of context and response: (1) identify important information in context for response, (2) model the relationship between context and response, (3) model the relationship between utterances in context.

Sometimes the relevance intensity of utterances in different conversations may be quite different, especially for open domain conversations. Contexts with different relevance intensity often have different requirement of relevance between the context and a proper response. So, we think the last one of those challenges is the most important for this task. To tackle these challenges, we propose a model named RCMN using the self-matching information in context to add a local relevance threshold for matching. In addition, we also adopt an existing model named SMN [11]. To the best of our knowledge, SMN is state-of-the-art model for multi-turn conversations matching. SMN focus on the relevance between response and each utterances of context and considers the matching information in both word level and sentence level. We finally ensemble the two models to predict. In Sect. 4, we will introduce our method in detail. Experiments in Sect. 5 show that RCMN is a comparable model to SMN. And because the two models have different significant diversity, model ensemble has achieved good improvement. Experiments details will be introduced in Sect. 5.

2 Related Work

As mentioned, this task can be regarded as a matching problem. Matching tasks can be divided into single turn conversation matching task and multi-turn conversation matching task. And the task is closer to multi-turn conversation matching.

For single turn text matching task, there are several notable works: Huang et al. (2013) propose a neural network structure which use word hashing strategy and full-connection feed-forward neural network for text matching [3]. This is an early application of neural networks to text matching. Hu et al. (2014) adopt convolutional neural network for the representation of query and response, propose two neural network structures: ARC-I and ARC-II [2]. Wan et al. (2016) adopt bidirectional LSTM for query and responses representation respectively and explore three different interaction functions for modeling matching signals between query and response [9]. Yang et al. (2016) propose a value shared weight strategy and a question attention network for text matching [13]. Wan et al. (2016) propose 2D-GRU for accumulating matching information in word interaction tensor [10]. Xiong et al. (2017) adopt kernel pooling for dealing with interaction tensor [12].

Recently, researchers begin to pay attention to multi-turn conversations matching. Lowe et al. (2015) concatenated the utterances of context and then treated multi-turn matching as single turn matching [6]. Zhou et al. (2016) propose a multi-view model including an utterance view and a word view to improve multi-turn response selection [14]. Wu et al. (2017) match a response with each utter-ance at first and accumulate matching information instead of sentences by a GRU, thus useful information for matching can be sufficiently retained [11].

3 Problem Formalization

In this task, the training data consists of raw multi-turn conversations. In the testing data, there are 10 candidates for each dialogue session. Among candidates for each session, only one reply is the ground truth while other candidates are randomly sampled from the data sets. We can abstract the retrieval problem as follow:

Assume that we have a data set $D = \{c_i, r_i, y_i\}_{i=1}^N$. c_i represents the context of conversation. Each c_i consists of k_i utterances: $c_i = \{u_1, u_2, ..., u_k\}_i$. r_i represents a candidate response of context. y_i represents whether r_i is ground truth for c_i. $y_i = 1$ means r_i is a proper response for c_i, otherwise $y_i = 0$. Thus, our goal is to learning a model M, for each pair of c_i and r_i, $M(c_i, r_i)$ given a matching degree between them. When $y_i = 1$, $M(c_i, r_i)$ is expected to output a value that as close to 1 as possible, and the situation is opposite when $y_i = 0$. Therefore, we can transform this matching task into a classification task during the training process. And In the prediction process, for each c_i, we use M to measure the matching degree for c_i and all the candidate responses of c_i, then we choose the r_j in candidate responses set $\{r_1, r_2, ..., r_t\}$ with the highest matching degree as the correct response.

Note that testing data has no y_i for sample i. And the training data consists of raw conversation so we should transform it into the form of D. We will introduce how to process the training data in Sect. 5.

4 System Description

As mentioned, our system adopted two deep neural networks and ensemble them for predicting. In this Section, we will first introduce RCMN we proposed in detail. And then we will introduce SMN briefly. Finally, we will introduce how we ensemble the two models.

4.1 Relevance Consistency Matching Network

As mentioned, sometimes the relevance intensity of utterances in different conversations may be quite different. Assume that there are two conversations, one has strong relevance between each two utterances of context denoted as c_1, the other has weak relevance denoted as c_2, the proper responses for c_1 and c_2 are r_1 and r_2 respectively. If there is a model that not consider the relevance intensity in context, the model is likely to predict that $M(c_1, r_1)$ is close to 1 but $M(c_2, r_2)$ is close to 0, but the right output is both 1. To solve this problem in RCMN, we consider to use the self-matching information in context. So, we first use RNN [8] to get sentence level representation of utterances in context and response. Then we let each two utterances in context and response do an interaction, then we can get an interaction tensor. Hence, the interaction tensor contains information of context relevance intensity and relevance between response and each utterance in context. Considering the outstanding performance of convolutional neural network in image processing and pattern recognition, we employ CNN [5] for extracting local features in interaction tensor into high level representation of matching features which is named final matching vector. To transfer the final matching vector into matching score, we adopt a Multilayered perception which can do a nonlinear mapping. The architecture of RCMN is shown in Fig. 1.

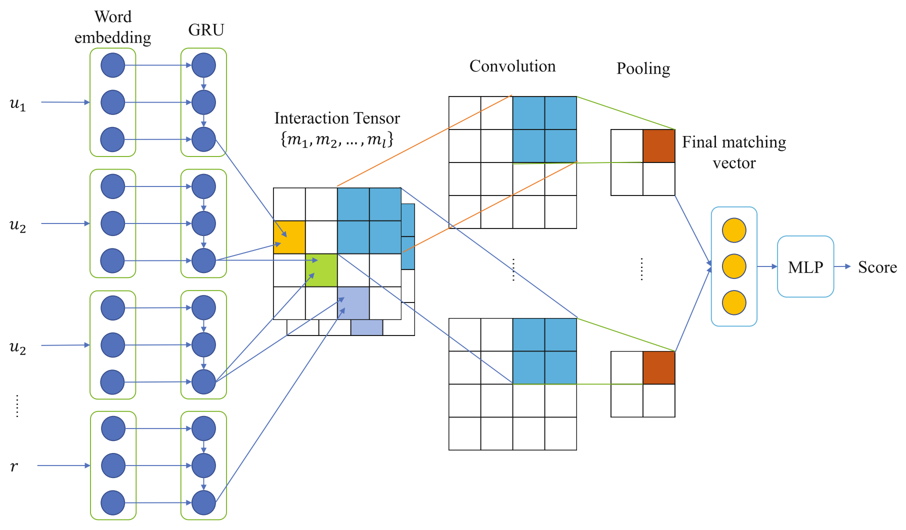

Fig. 1. The architecture of RCMN.

RCMN consists of 4 parts: Sentence RNN Layer, Global Interaction Layer, Convolution and Pooling Layer, Output Layer.

Sentence RNN Layer transfer utterances and response with word embedding representation into a sentence level representation. Each utterance and response will be encoded into one vector. We use the last state of GRU [1] as output. Then the Global Interaction Layer use these vectors to generate interactions of each (u, r) and (u, u) pair. We denote Sentence RNN Layers output as:

$$S = [u_1, u_2, \ldots, u_k, r] \tag{1}$$

In this layer, we use a series of matrixes denoted as $[w_1, w_2, \ldots, w_l]$ as interaction weight. For w_i, we can get an interaction matrix m_i, m_i is computed as follow:

$$m_i = S^T w_i S \tag{2}$$

After Global Interaction Layer, we will get l interaction matrixes which can also be called interaction tensor. The interaction tensor is the input of Convolution and Pooling Layer which contains a convolution operation and a max pooling operation. Each m_i will be a channel of convolution. The output of Convolution and Pooling Layer will be concatenated into one vector named final matching vector. Finally, Output Layer which is a Multilayer perceptron use the final matching vector to calculate the final matching score between context and response.

4.2 Sequential Matching Network

For better modeling the relationship of response and each utterance in context, we adopt SMN which consider the matching information in both word level and sentence level as a supplement. It consists of three parts: Utterance-Response Matching Layer, Matching Accumulation Layer, Matching Prediction Layer.

Utterance-Response Matching Layer used both CNN and RNN. It concerned the matching information between each utterance of context and response. And it considered both word level and sentence level matching information. After Utterance-Response Matching Layer, we got some single turn matching vectors for each utterance of context and response pair. Then Matching Accumulation Layer used GRU to model the matching information of the entire conversation, by letting those single turn matching vectors go through this GRU. Finally, Matching Prediction Layer used dense layers transfers the output of Matching Accumulation Layer into matching degree.

Due to space limitation, we will not discuss SMN further more. See [11] for more details.

4.3 Model Ensemble

We use the weighted average of the results of each model as an ensemble result. To get a better ensemble result, weight of each model should be positively related to models performance. We denote the weight of M_i using $train_d$ to train as $w^d_{m_i}$, the data set d as $d = \{train_d, val_d, test_d\}$, the precision of M_i on val_d when using $train_d$ for training as $p^d_{m_i}$. For convenience, we set the $w^d_{m_i}$ as follow:

$$w^d_{m_i} = \frac{p^d_{m_i}}{\sum_j^L (\sum_d^H p^d_{m_j})} \tag{3}$$

where L means the number of models, H means the number of validation data sets. In this paper, L is equal to 2. We denote the result of M_i on sample q when use $train_d$ for training as $r^d_{m_i}$. Then the ensemble result ER of q can be formalized as follow:

$$ER_q = \frac{\sum_{i=1}^L (\sum_{d=1}^H w^d_{m_i} \cdot r^d_{m_i})}{L \cdot H} \tag{4}$$

5 Experiments

Our code is available at https://github.com/jimth001/NLPCC2018_Multi_Turn_Response_Selection. It is implemented using python 3.6. Main external packages we used are TensorFlow[1], thulac[2], word2vec[3], NumPy[4].

5.1 Data Sets and Metrics

We used the data sets provided by NLPCC 2018. Table 1 gives the statistics.

[1] https://tensorflow.google.cn/
[2] http://thulac.thunlp.org/
[3] https://pypi.org/project/word2vec/
[4] http://www.numpy.org/

Table 1. Statistics of origin data sets.

	Train	Test
Number of contexts	5000K	10K
Positive responses per context	1	1
Negative responses per context	-	9
Average turns of context	3.10	3.10
Average length of utterance	10.18	10.77

Participants are required submit the index of the ground truth they inferred for each conversation. The evaluation metric is precision of retrieved results. We give a formalization description as follows:

$$percision = \frac{\sum_i^N I(\arg\min_j(M(c_i, r_{i_j})) = \arg\min_k(y_{i_k}))}{|D|} \tag{5}$$

$$r_{i_j} \in \{r_1, r_2, ..., r_t\}_i \tag{6}$$

$$y_{i_k} \in \{y_1, y_2, ..., y_t\}_i \tag{7}$$

where $\{r_1, r_2, ..., r_t\}_i$ are candidate responses of c_i, $\{y_1, y_2, ..., y_t\}_i$ are the labels for $\{r_1, r_2, ..., r_t\}_i$. $I(\cdot)$ is an indicator function.

5.2 Experiments on Training Data

In the training and validation process, we shuffle the training data provided by organizer and partition it into 8 folds. We choose one-fold as validation set, one-fold as testing set, and others as training set. And we use random sampling to select 9 negative responses for each conversation in validation and testing set. We denote the validation set, training set and testing set as val_1, $train_1$ and $test_1$ respectively. For further experiments, we use the same method to generate val_2, $train_2$ and $test_2$. Table 2 gives the statistics. We use validation set to select hyper parameters, and use testing set to test the effect of our models.

Table 2. Statistics of data sets divided from training set.

	$train_1$	val_1	$test_1$	$train_2$	val_2	$test_2$
Number of contexts	3750K	625K	625K	3750K	625K	625K
Positive responses per context	1	1	1	1	1	1
Negative responses per context	-	9	9	-	9	9
Average turns of context	3.10	3.09	3.10	3.10	3.10	3.09
Average length of utterance	10.59	10.61	10.57	10.59	10.58	10.59

Because the training data only has raw conversations, which means that there is no (c, r, y) pairs in the data. So, we manually construct these pairs using the following strategy: For each conversation in training data, take the last turn as r, other as c, and the y is 1. In the training process, for each c, random sampling 2 sentences in the whole data set as r, and the corresponding y is 0.

Our preprocessing of text is simple. We selected thulac as our tokenizer and removed stop words using default stop words list in thulac. We adopted word2vec [7] to generate word embedding. We set main parameters of thulac as follows: user_dict is None, T2S is True, seg_only is True, filt is True. And main parameters of word2vec are set as follows: size is 200, window is 8, sample is 1e-5, cbow is 0, min_count is 6. Parameters not mentioned are set as default value. Only $train_1$ is used for pre-train word embedding, and all of our experiments is based on it. The word embedding is not trainable in both models.

Finally, we set some same parameters for both two models: the maximum context length is 10, the utterance length is 50. We padded zeros if context length and utterance length is less than 10 and 50 respectively. If context length and utterance length is more than 10 and 50, we chose the last 10 and 50 respectively. The parameters were updated by stochastic gradient descent with Adam algorithm [4] on a single 1080Ti GPU. The initial learning rate is 0.001, and the parameters of Adam, $\beta 1$ and $\beta 2$ are 0.9 and 0.999 respectively. We adopted cross entropy loss for training.

We finally set SMN parameters following [11]: the dimensionality of the hidden states of GRU in Utterance-Response Matching Layer is 200, window size of convolution and pooling is $(3, 3)$, the dimensionality of the hidden states of GRU in Matching Accumulation Layer is 50.

We finally set parameters of RCMN as follows: the dimensionality of the hidden states of GRU is 200, the shape of interaction weight is $(200, 2, 200)$, window size of convolution and pooling is $(3, 3)$, the number of filters is 8, the dimensionality of the hidden layer of MLP is 50.

The experiments result based on training data released by organizer is shown in Table 3.

5.3 Experiment Result on Testing Data and Analysis

We used $train_1$ and $train_2$ to train RCMN and SMN. And ensemble these models using the method introduced in Sect. 4. The official ranking results on test data sets are summarized in Table 4.

We conducted ablation experiments on official test data to examine the usefulness of models and training data sets. Experiment result is shown in Table 5. We can see that each model and data set contribute to the final result. The contribution of $train_1$ is more than the contribution of $train_2$. The contribution of RCMN is more than the contribution of SMN. Result of single model on official test data is shown in Table 6.

Experiments shown in Tables 3 and 5 show that RCMN is a comparable model to state-of-the-art model for multi-turn conversation response selection. Table 3 also shows that RCMN and SMN are insensitive to data not seen on this data set, because that the performance has not significant improvement when training data and validation data have overlap. For example, $train_1$ and val_1

Table 3. Experiments result on training data.

Training set	Model name	Testing set	Precision
$train_1$	SMN	val_1	60.10
$train_2$	SMN	val_1	58.80
$train_1$	RCMN	val_1	59.00
$train_2$	RCMN	val_1	59.56
$train_1$	RCMN+SMN	val_1	61.91
$train_2$	RCMN+SMN	val_1	62.58
$train_1$ and $train_2$	RCMN+SMN	val_1	63.48
$train_1$	SMN	val_2	60.64
$train_2$	SMN	val_2	58.96
$train_1$	RCMN	val_2	59.81
$train_2$	RCMN	val_2	58.91
$train_1$	RCMN+SMN	val_2	62.66
$train_2$	RCMN+SMN	val_2	62.27
$train_1$ and $train_2$	RCMN+SMN	val_2	63.64
$train_1$	SMN	$test_1$	60.17
$train_2$	SMN	$test_1$	58.81
$train_1$	RCMN	$test_1$	59.08
$train_2$	RCMN	$test_1$	59.67
$train_1$	RCMN+SMN	$test_1$	62.02
$train_2$	RCMN+SMN	$test_1$	62.69
$train_1$ and $train_2$	RCMN+SMN	$test_1$	63.55
$train_1$	SMN	$test_2$	60.69
$train_2$	SMN	$test_2$	58.86
$train_1$	RCMN	$test_2$	59.82
$train_2$	RCMN	$test_2$	58.92
$train_1$	RCMN+SMN	$test_2$	62.72
$train_2$	RCMN+SMN	$test_2$	62.23
$train_1$ and $train_2$	RCMN+SMN	$test_2$	63.69

Table 4. Official results on test data.

System name	Precision
ECNU	62.61
wyl_buaa	59.03
Yiwise-DS	26.68
laiye_rocket	18.13
ELCU_NLP	10.54

Table 5. Evaluation results of model ablation.

Models and data sets	Precision	Loss of performance
All	59.03	-
– SMN	57.51	1.52
– SMN on $train_1$	58.84	0.19
– SMN on $train_2$	58.86	0.17
– RCMN	56.41	2.61
– RCMN on $train_1$	58.22	0.81
– RCMN on $train_2$	58.68	0.35
– $train_1$	57.44	1.59
– $train_2$	58.33	0.70

Table 6. Single model on official test data.

Training set	Model name	Precision
$train_1$	SMN	56.11
$train_2$	SMN	53.24
$train_1$	RCMN	55.92
$train_2$	RCMN	55.58

have no overlap but $train_1$ and val_2 have. Model Ablation shows that RCMN slightly outperform SMN on official testing data. We think that RCMN does not obviously outperform SMN on NLPCC 2018 data set because RCMN does not consider the matching information in word level and the relevance intensity of conversations in the data set is not quite different. Table 6 shows that SMN using $train_2$ for training is weaker than RCMN using $train_1$ or $train_2$ for training on official testing data. This is probably because $train_2$ is more different on relevance intensity of conversations from official testing data. If so, this is also a proof of that RCMN can adapt to different conversations having different relevance intensity. Ensemble result shows that model ensemble brings good improvement. We think its because of the significant diversity of the two models.

6 Conclusion

In this paper, we proposed RCMN which considered self-matching information in context for response selection. RCMN focus on better modeling the relevance consistency of conversations but lacks in considering word level matching information. We also employed an existing model named SMN. Experiments in Sect. 5 show that RCMN is a comparable model to SMN, which is regarded as state-of-the-art model for multi-turn conversation response selection. We also proposed an ensemble strategy for the two models. And experiments show that ensemble of models makes good improvement. We have given some reasonable analysis for experiments result in Sect. 5. The official results show that our solution takes 2nd place. In feature work, we will pay more attention to capture more matching information in single turn matching and to model relevance intensity more effectively.

Acknowledgments. This work was supported in part by the Natural Science Foundation of China (Grand Nos. U61672081, 1636211, 61370126), and Beijing Advanced Innovation Center for Imaging Technology (No. BAICIT-2016001) and National Key R&D Program of China (No. 2016QY04W0802).

References

1. Chung, J., Gulcehre, C., Cho, K.H., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. Eprint Arxiv (2014)
2. Hu, B., Lu, Z., Li, H., Chen, Q.: Convolutional neural network architectures for matching natural language sentences. Adv. Neural Inf. Process. Syst. **3**, 2042–2050 (2015)
3. Huang, P.S., He, X., Gao, J., Deng, L., Acero, A., Heck, L.: Learning deep structured semantic models for web search using clickthrough data. In: ACM International Conference on Conference on Information & Knowledge Management, pp. 2333–2338 (2013)
4. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. Computer Science (2014)
5. Lecun, Y., et al.: Backpropagation applied to handwritten zip code recognition. Neural Comput. **1**(4), 541–551 (2014)
6. 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)
7. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. Adv. Neural Inf. Process. Syst. **26**, 3111–3119 (2013)
8. Rumerlhar, D.E.: Learning representation by back-propagating errors. Nature **323**(3), 533–536 (1986)
9. Wan, S., Lan, Y., Guo, J., Xu, J., Pang, L., Cheng, X.: A deep architecture for semantic matching with multiple positional sentence representations, pp. 2835–2841 (2015)
10. Wan, S., Lan, Y., Xu, J., Guo, J., Pang, L., Cheng, X.: Match-SRNN: modeling the recursive matching structure with spatial RNN. Comput. Graph. **28**(5), 731–745 (2016)
11. Wu, Y., et al.: Sequential matching network: a new architecture for multi-turn response selection in retrieval-based chatbots. In: Meeting of the Association for Computational Linguistics, pp. 496–505 (2017)
12. Xiong, C., Dai, Z., Callan, J., Power, R., Power, R.: End-to-end neural ad-hoc ranking with kernel pooling, pp. 55–64 (2017)
13. Yang, L., Ai, Q., Guo, J., Croft, W.B.: aNMM: ranking short answer texts with attention-based neural matching model. In: ACM International on Conference on Information and Knowledge Management, pp. 287–296 (2016)
14. Zhou, X., et al.: Multi-view response selection for human-computer conversation. In: Conference on Empirical Methods in Natural Language Processing, pp. 372–381 (2016)