

HFL-RC System at SemEval-2018 Task 11: Hybrid Multi-Aspects Model for Commonsense Reading Comprehension

Zhipeng Chen^{*†}, Yiming Cui^{*†}, Wentao Ma[†], Shijin Wang[†], Ting Liu[‡] and Guoping Hu[†]

[†]Joint Laboratory of HIT and iFLYTEK (HFL), iFLYTEK Research, Beijing, China

[‡]Research Center for Social Computing and Information Retrieval (SCIR),

Harbin Institute of Technology, Harbin, China

[†]{zpchen, ymcui, wtma, sjwang3, gpku}@iflytek.com

[‡]tliu@ir.hit.edu.cn

Abstract

This paper describes the system which got the state-of-the-art results at SemEval-2018 Task 11: Machine Comprehension using Commonsense Knowledge. In this paper, we present a neural network called Hybrid Multi-Aspects (HMA) model, which mimic the human's intuitions on dealing with the multiple-choice reading comprehension. In this model, we aim to produce the predictions in multiple aspects by calculating attention among the text, question and choices, and combine these results for final predictions. Experimental results show that our HMA model could give substantial improvements over the baseline system and got the first place on the final test set leaderboard with the accuracy of 84.13%.

1 Introduction

Machine Reading Comprehension (MRC) has become a spotlight topic in recent natural language processing field. MRC consists of various sub-tasks, such as cloze-style reading comprehension (Hermann et al., 2015; Hill et al., 2015; Cui et al., 2016, 2018), span-extraction reading comprehension (Rajpurkar et al., 2016) and open-domain reading comprehension (Chen et al., 2017), etc. One key problem in reading comprehension is that how the machine utilizes the commonsense knowledge for real-life reading comprehension. In the SemEval-2018 Task 11: Machine Comprehension using Commonsense Knowledge (Osternann et al., 2018), the organizers provide narrative texts about everyday activities and require the participants to build a system for answering questions based on this text. To tackle this problem, in this paper, we present a novel model called Hybrid Multi-Aspects (HMA) model. The main features of our model can be concluded as follows.

- Our model is mainly based on the neural network approach without using any external knowledge, such as script knowledge, etc.
- We aim to produce the predictions in multiple aspects by calculating attention among the text, question and choices, and combine these results for final predictions.
- We add additional features on the embedding representations for the text, question and choices, including word matching feature and part-of-speech tags, etc.

2 System Description

We will first give a brief introduction of the SemEval-2018 Task 11. Then the pre-processing and the proposed Hybrid Multi-Aspects model will be illustrated afterwards. A quick glance of the neural architecture of the HMA model is depicted in Figure 1.

2.1 Task Description

Given a short context about the narrative texts about everyday activities and several following questions about the context, the participants are required to build a system to solve the question by choosing the correct answer from two choice choices. The participants are encouraged to use external knowledge to improve their systems and there is no restrictions. For more details, please refer Osternann et al. (2018).

2.2 Pre-processing

We describe the pre-processing procedure on the evaluation data, which can be listed as follows.

1. All punctuations are removed.
2. All words are lower-cased.

^{*}Equal contribution.

- All sentences are tokenized by Natural Language Toolkit (NLTK) (Bird et al., 2009).

2.3 Embedding Layer

In this layer, we aim to project text, question and choices into embedding representations. The final embedding representations are composed by three components, which can be listed as follows.

- Word embedding:** We use pre-trained GloVe embedding (Pennington et al., 2014) for word representations, whose size is 100d (d for dimension).
- Char embedding:** We use randomly initialized embedding matrix for char-level embeddings, whose size is 8d. We use 1D-convolution operation with filter length of 5 and output size of 100d. Then we apply max-over-time-pooling to obtain the final representation, whose size is 100d.
- Feature embedding:** We also adopt several hand-crafted features for enhancing the word representations. In this paper, we adopt three features which can be illustrated as follows.
 - Part-of-Speech:** We use NLTK (Bird et al., 2009) for part-of-speech tagging for each word in the text, question and choices. In this paper, we assign different trainable vectors with size of 16d for each part-of-speech tag.
 - Word matching:** Taking the text as an example, if the text word appear in question or choice, we set this feature as 1. If not, set it as 0. In this way, we can also add this feature to the question and choice.
 - Word fuzzy matching:** Similar to the ‘word matching’ feature, but we loosen the matching criteria as partial matching. For example, we regard ‘teacher’ and ‘teach’ as fuzzy matching, because the string ‘teacher’ is partially matched by ‘teach’.

After obtaining three parts of the embeddings, the final representation is the concatenation of three embeddings, forming the size of 100d+100d+16d+2d=218d.

2.4 RNN Layer

After obtaining the embedding representations E , we first feed this into a shared Highway network (Srivastava et al., 2015) across the text, question and choices. The output activation is chosen as tanh.

$$HW = \tanh(\text{Highway}(E)) \quad (1)$$

After highway network, we get text, question and choices’ presentation $HW_T \in \mathbb{R}^{t \times e}$, $HW_Q \in \mathbb{R}^{q \times e}$ and $HW_C \in \mathbb{R}^{cn \times c \times e}$, where t , q , c and cn represent the text max length, question max length, choice max length and the number of choices respectively. Then we use Bi-directional LSTM (Graves and Schmidhuber, 2005) and concatenate the forward and backward hidden representations to obtain the contextual representations of text $B_T \in \mathbb{R}^{t \times h}$, question $B_Q \in \mathbb{R}^{q \times h}$ and choices $B_C \in \mathbb{R}^{cn \times c \times h}$, where h presents Bi-LSTM hidden size. Note that, we use separated Bi-LSTM (without sharing weights) for the text, question and choices.

$$B_T = \text{Bi-LSTM}(HW_T) \quad (2)$$

$$B_Q = \text{Bi-LSTM}(HW_Q) \quad (3)$$

$$B_C = \text{Bi-LSTM}(HW_C) \quad (4)$$

2.5 Attention Layer

After obtaining the contextual representations of the text, question and choices, we will calculate the attentions between different combinations in order to characterize the choices in multiple aspects. In this paper, we aim to obtain three representations

- choice-aware text representation H_{CT}
- choice-aware question representation H_{CQ}
- self-attended question representation H_{QQ}

First, we will calculate choice-aware text representation H_{CT} to extract the choice-relevant part from text representation. Following Cui et al. (2017), we first calculate dot similarity between each word in the text and choice to obtain the matching matrix. Then we apply row-wise softmax to obtain individual text-level attention with respect to each choice word, denoted as $M_{CT} \in \mathbb{R}^{c \times t}$.

$$M'_{CT} = B_C \cdot B_T^T \quad (5)$$

$$M_{CT} = \text{softmax}(M'_{CT}) \quad (6)$$

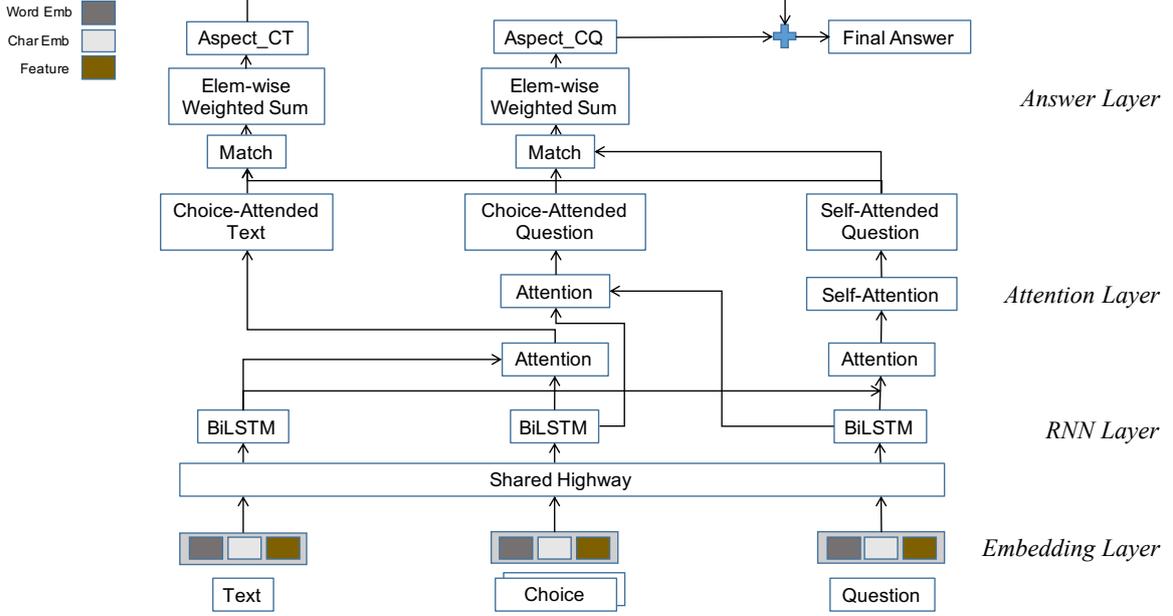


Figure 1: Neural architecture of our Hybrid Multi-Aspects (HMA) model. Note that, there are two choices processed in parallel.

Then we will get choice-aware text representation $H_{CT} \in \mathbb{R}^{c \times 2h}$. Note that, we will also concatenate choice LSTM representation B_C to enhance the representation power.

$$H'_{CT} = M_{CT} \cdot B_T \quad (7)$$

$$H_{CT} = \text{concat}[H'_{CT}; B_C] \quad (8)$$

In the same way, we can also obtain the choice-aware question representation $H_{CQ} \in \mathbb{R}^{c \times 2h}$ and the question-aware text representation $H_{QT} \in \mathbb{R}^{q \times 2h}$. However, in order to get further extract question information, we apply additional self-attention procedure to obtain self-attention matching matrix $M_{QQ} \in \mathbb{R}^{q \times q}$.

$$M'_{QQ} = H_{QT} \cdot H_{QT}^T \quad (9)$$

$$M_{QQ} = \text{softmax}(M'_{QQ}) \quad (10)$$

Then we will get self-attended question representation $H_{QQ} \in \mathbb{R}^{q \times 2h}$.

$$H'_{QQ} = M_{QQ} \cdot B_Q \quad (11)$$

$$H_{QQ} = \text{concat}[H'_{QQ}; B_q] \quad (12)$$

2.6 Answer Layer

In this module, we will utilize the multi-aspect representations from previous layer to get a hybrid prediction. First, we will calculate the similarity between the choice-aware text representation H_{CT} and self-attended question representation H_{QQ} to obtain the deep matching matrix

D_{CT} . Similarly, we can also calculate the similarity between the choice-aware question representation H_{CQ} and self-attended question representation H_{QQ} to obtain the deep matching matrix D_{CQ} .

$$D_{CT} = H_{CT} \cdot H_{QQ}^T \quad (13)$$

$$D_{CQ} = H_{CQ} \cdot H_{QQ}^T \quad (14)$$

Then we will apply a element-wise weight $W_{CT} \in \mathbb{R}^{c \times q}$ and get the weighted sum of D_{CT} and output a scalar value. Note that, we have two choices, so the final output should be $A_{CT} \in \mathbb{R}^{1 \times 2}$. In the same way, we can also calculate $A_{CQ} \in \mathbb{R}^{1 \times 2}$.

$$A_{CT} = \sum D_{CT} \odot W_{CT} \quad (15)$$

$$A_{CQ} = \sum D_{CQ} \odot W_{CQ} \quad (16)$$

Finally, we apply softmax function to A_{CT} and A_{CQ} and sum the probabilities to get final predictions A .

$$A = \text{softmax}(A_{CT}) + \text{softmax}(A_{CQ}) \quad (17)$$

2.7 Training Criterion

We use categorical cross entropy to calculate loss between the predicted answer probability A and real answer.

3 Experiments

3.1 Experimental Setups

We listed the main hyper-parameters of our model in Table 1. The word embeddings are initialized by the pre-trained GloVe word vectors (Common Crawl, 6B tokens, 100-dimension) (Pennington et al., 2014). The words that do not appear in the pre-trained word vectors are set to the ‘unk’ token and initialized accordingly. We use Adam for weight optimizations with default parameters. The models are built on Keras (Chollet, 2015) with Theano backend (Theano Development Team, 2016). We choose our model by the performance of the development set.

Symbol	Descriptions	#
t	Text max length	300
q	Question max length	20
cn	Number of choices	2
e	Embedding size	218
h	Bi-LSTM output size	200
c	choice max length	10

Table 1: Hyper-parameter settings of our system.

3.2 Results

The experimental results are shown in Table 2. We also listed some of the top-ranked systems in this evaluation¹. As we can see that our single model could give substantial improvements over Simple RNN baseline system by 12.78% on development set. For further improve the performance, we carried out 7-ensemble by voting approach produced by each single system. The results show that the ensemble system could give further improvements where 1.98% and 3.19% gains on development and test set respectively. When compared to the several top-ranked systems, our HMA model surpasses all the competitors and got on the first place in SemEval-2018 Task 11.

4 Discussion

To have a better understanding of the data, we divided the data into different question types, which can be seen in Figure 2. As we can see that the yes/no question takes up 27% proportion, which

¹Full SemEval-2018 Task 11 leaderboard can be accessed through: <https://competitions.codalab.org/competitions/17184#results>

Model	Dev	Test
Simple RNN	71.70%	-
HMA Model (single)	84.48%	80.94%
HMA Model (ensemble)	86.46%	84.13%
Yuanfudao (BananaTree)	-	83.95%
MITRE (guidoz)	-	82.27%
(jiangnan)	-	80.91%
Rusalka (minerva)	-	80.48%
SLQA (mingyan)	-	79.94%

Table 2: Experimental results. Top-ranked participant’s systems are also included.



Figure 2: Proportions of different question types.

is quite different from most of the other reading comprehension datasets. The yes/no questions require better handling the negation and deeper understanding in question.

Another observation is that, different from SQuAD dataset (Rajpurkar et al., 2016), the choices are not extracted from the text but written by the human. So there are many questions that the ordinary word matching failed to give correct answer. For example, the example shown in Figure 3, the choice ‘dirty’ is not exactly match the word ‘dirt’ in passage, which add difficulties in solving these problems. In our model, we add additional partial matching feature to indicate the underlying relations. The final results show that adding partial matching feature could give an improvement of 1%~2% in accuracy, indicating that the feature is helpful for the model to identify the words that have similar meanings. Also, we have tried to use word stemming to restore the word to

Passage
My shower was filthy and grimy. Whoa! All that shower dirt had to go. So what I did was, I went over to my local Walmart, and I bought a sponge, a pair of rubber gloves, any empty five gallon bucket, bleach and shower spray cleaner. I gathered those things and bought them all, and all cost me 9.89 in total, including tax for these shower cleaning supplies. I headed back home, filled a bucket up with two gallons of hot water, and added two tablespoons of bleach and about a quarter cup of shower cleaner. After that I put on some rubber gloves onto my hands in order to protect them from the harsh chemicals. I then put the sponge deep into the bucket cleaning solution, and then proceeded to scrub my shower with the dousing wet sponge. I rubbed out all of the grime out of the shower with the wet, chemically treated sponge. After I completed the scrub down, it was time to rinse, so I simply ran the shower head with lukewarm water in order to drain all the grime down the shower drain. After that, my shower was finally clean, grime free and looked like new!
Question
Why did they spray so much of the cleaner?
Candidates
A: Dirty B: Clean

Figure 3: Example of the partial matching problem.

its stemming form, we observed a significant drop in the final performance, suggesting that a much powerful model is needed to further tackle these problems.

5 Conclusion

In this system description, we propose a novel neural network system called Hybrid Multi-Aspects (HMA) model for the SemEval-2018 Task 11. In this model, we aim to produce multi-aspect output and combine them for final predictions. We adopt a simple dot product to measure the similarity between the text, question and choices. We also enhanced the question representations by using self-attention mechanism. The final predictions are obtained by accumulating the probabilities from various aspects. The final leaderboard provided by the task officials shows that the proposed HMA model got the first place among 24 participants with the accuracy of 84.13%. In the future, we would like to investigate how to effectively adopt the external knowledge into machine reading comprehension model and would like to focus on solving the questions that is less likely answered by statistical data.

Acknowledgments

This work was supported by the National 863 Leading Technology Research Project via grant 2015AA015409.

References

Bird, Steven, Edward Loper, and Ewan Klein. 2009. *Natural Language Processing with Python*.

O'Reilly Media Inc.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. *Reading wikipedia to answer open-domain questions*. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 1870–1879. <https://doi.org/10.18653/v1/P17-1171>.

François Chollet. 2015. Keras. <https://github.com/fchollet/keras>.

Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. 2017. *Attention-over-attention neural networks for reading comprehension*. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 593–602. <https://doi.org/10.18653/v1/P17-1055>.

Yiming Cui, Ting Liu, Zhipeng Chen, Wentao Ma, Shijin Wang, and Guoping Hu. 2018. Dataset for the first evaluation on chinese machine reading comprehension. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA).

Yiming Cui, Ting Liu, Zhipeng Chen, Shijin Wang, and Guoping Hu. 2016. Consensus attention-based neural networks for chinese reading comprehension. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. Osaka, Japan, pages 1777–1786.

Alex Graves and Jürgen Schmidhuber. 2005. Frame-wise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks* 18(5-6):602–610.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*. pages 1684–1692.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2015. The goldilocks principle: Reading children’s books with explicit memory representations. *arXiv preprint arXiv:1511.02301*.

Simon Ostermann, Michael Roth, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. 2018. SemEval-2018 Task 11: Machine Comprehension using Commonsense Knowledge. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*. New Orleans, LA, USA.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. *Glove: Global vectors for word representation*. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language*

Processing (EMNLP). Association for Computational Linguistics, pages 1532–1543. <http://aclweb.org/anthology/D14-1162>.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 2383–2392. <https://doi.org/10.18653/v1/D16-1264>.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv:1505.00387*.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints* abs/1605.02688. <http://arxiv.org/abs/1605.02688>.