Exploiting Explicit Matching Knowledge with Long Short-Term Memory

Xinqi Bao and Yunfang Wu (✉)

Key Laboratory of Computational Linguistics (Peking University), Ministry of Education
School of Electronic Engineering and Computer Science, Peking University
wuyf@pku.edu.cn

Abstract. Recently neural network models are widely applied in text-matching tasks like community-based question answering (cQA). The strong generalization power of neural networks enables these methods to find texts with similar topics but miss detailed matching information. However, as proven by traditional methods, the explicit lexical matching knowledge is important for effective answer retrieval. In this paper, we propose an ExMaLSTM model to incorporate the explicit matching knowledge into the long short-term memory (LSTM) neural network. We extract explicit lexical matching features with prior knowledge and then add them to the local representations of questions. We summarize the overall matching status by using a bi-directional LSTM. The final relevance score is calculated using a gate network, which can dynamically assign appropriate weights to the explicit matching score and the implicit relevance score. We conduct extensive experiments for answer retrieval in a cQA dataset. The results show that our proposed ExMaLSTM model outperforms both the traditional methods and various state-of-the-art neural network models significantly.

Keywords: lexical matching knowledge, LSTM, question answering

1 Introduction

The community-based question answering (cQA) attracts considerable attention in recent years. Traditional question answering systems, driven by evaluations such as the Text REtrieval Conference (TREC), generally aim to retrieve short and factoid answers. But questions from cQA services tend to be more subjective and complex, and the answers are often in a causal style, including both fact description and subjective opinions. So the answer retrieval task in cQA is more challenging.

Traditional methods on cQA retrieval are mainly based on surface lexical matching, which suffer from the severe lexical gap problem. Recently, researchers have proposed various neural networks and semantic embedding based methods to overcome this problem (for example, Hu et al., 2014; Palangi et al., 2015; Zhou et al., 2015; Qiu and Huang, 2015), which take advantage of the strong generalization power of neural networks. Generally speaking, these methods try to dive into the latent embedding space and then calculate the relevance score to find the pairs which are mostly like to match each other.

However, there are limitations for most of previous neural network methods in practice. First, a large amount of training data is required to learn appropriate parameters, which is unrealistic for some specified domains. Second, there exist out of vocabulary (OOV) words and unseen phrases, and it is hard to embed their latent semantics. Third, the strong generalization power enables these methods to find texts with similar topics, but they may miss or obscure the detailed matching information, so underestimate the relevance of those text spans with explicitly matched points.

Table 1 shows an example. The basic neural network model of this paper successfully captures the "delicious food" topic but loses the explicitly matched key point "spicy hot pot", which is rare or even unseen in the training data but is the semantic focus of this question. We can see that the traditional method of direct lexical matching still has its value.

Table 1. An example of a question and its related answers. The unexpected answer is returned by the basic LSTM model of this paper; the expected answer is the right answer.

Question: I want to know where is the most delicious spicy hot pot in Beijing?
Unexpected Answer: Beijing is the culinary capital where roasted duck, sauteed noodles with vegetables and other local snacks are easy available. Just please walk on the Wangfujing Snack Street to spend happy time with various delicious foods. The address is ……
Expected Answer: On a cold winter day, you may like to have something hot with your family. Then the spicy hot pot is perhaps the best choice for you. Now let's introduce the most famous hot pots in Beijing below ……

In this paper, we focus on exploiting such explicit matching information in question-answer pairs for answer retrieval. We propose an ExMaLSTM model, which extends the traditional LSTM model as follows.

- We extract explicit lexical matching features of question-answer pairs with prior knowledge, by using rich language resources.
- We incorporate these explicit matching features into the original word vector for each word in the question. The overall explicit matching status is summarized by a bi-directional LSTM, and then the explicit matching score is calculated via the summarized representation.
- We calculate the final relevance score by using a gate network, which can dynamically assign different weights to the explicit matching score and the implicit relevance score. The implicit relevance score is calculated by the basic LSTM model of this paper.

We conduct extensive experiments for answer retrieval in a Chinese cQA dataset. The experimental results show that our extended ExMaLSTM model outperforms various state-of-the-art neural network models significantly. It can well capture the explicit lexical matching information and assign appropriate weights to explicit and implicit scores.

2 Related work

There are a lot of researches to utilize neural network based models in cQA retrieval. They can be clustered to the following two groups.

The first idea is to embed the question and answer separately into latent semantic spaces, and then calculate the implicit relevance score with embedded vectors. Studies include bag-of-words based embedding models (Wang et al., 2011), recursive neural network model (RNN) (Iyyer et al., 2014), convolutional neural network (CNN) model (Hu et al., 2014), long short-term memory network model (Palangi et al.,2015) and combined model (Zhou et al., 2015). Qiu and Hunag (2015) implemented a tensor transformation layer on CNN based embeddings to capture the interactions between question and answer more effectively.

The second idea is to conduct matching process with pairs of local embeddings and then calculate the overall relevance score. Works include enhanced lexical model (Yih et al., 2013), DeepMatch (Lu and Li, 2013). Pang et al. (2016) calculated word similarity matrix from pairs of words between question and answer, and then built hierarchical convolution layers on it. Yin and Schutze (2015) proposed MultiGranCNN, which integrates multiple matching models with different levels of granularity. Wan et al. (2016) proposed Multiple Positional Sentence Representation (MPSR), which uses LSTM and interactive tensor to capture matching points with positional local context. The difference with our work is that they still depend on embeddings of local information, thus cannot fully capture the explicit matching information of question-answer pairs.

Some other works try to incorporate non-textual information into the basic neural cQA model. Hu et al. (2013) used a deep belief network (DBN) to learn joint representations for textual features and non-textual features. Bordes et al. (2014) learnt joint embeddings of words and knowledge base constituents with subgraph embedding method.

To the best of our knowledge, most of the neural network models in cQA retrieval pay little attention to the explicit lexical matching information of text pairs. Wang and Nyberg (2015) simply combined their LSTM neural network model with the exact keyword matching score, but their method is quite different from our work in the following aspects. 1) They only extract the cardinal numbers and proper nouns to do keyword matching, while our work extracts plenty of lexical matching information. 2) They use the traditional Okapi BM25 algorithm to calculate the keywords matching score, while we employ a bi-directional LSTM network to predict the explicit matching status. 3) They use an external gradient boosting decision tree (GBDT) method to combine features, while we exploit a gate network to dynamically assign different importance weights to the implicit relevance score and explicit matching score.

3 The Basic model

We first describe the basic neural network model adopted in this paper for question-answer relevance calculation, which is depicted in Figure 1. We utilize a bi-directional LSTM to represent questions, and propose a Sent-LDA model to represent answers.

Then a three-way tensor is employed to model the interactions of question-answer pairs. Finally, a multilayer perception (MLP) is utilized to calculate the relevance score.

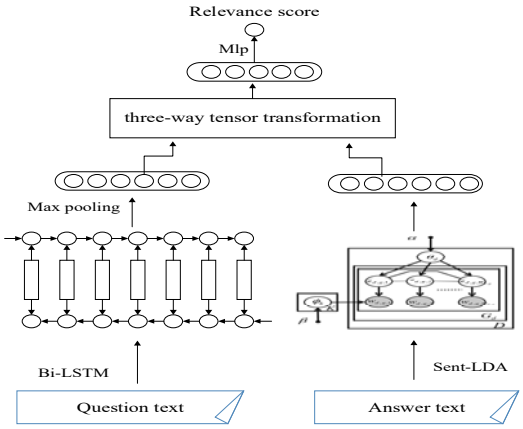

Fig. 1. The basic network model of this paper

3.1 Question Representation

Like Palangi et al. (2015), we use the bi-directional LSTM to embed questions into latent semantic representations, which can effectively capture the long-range dependencies of context information. We use max pooling through time to extract the final fixed-length representation for the question.

3.2 Answer Representation

We can also use LSTM to generate latent representations for answers. However, answers in a cQA forum often consist of multiple sentences and are much longer than questions, which makes the training process very time consuming. So, we utilize a sentence-level LDA (Sent-LDA), inspired by phrase-LDA (Kishky et al., 2014), to model sentence level information. It runs fast while achieves comparable results with neural based representations.

The Sent-LDA is the same with the classical LDA except that all words within a sentence are constrained to a unique topic. We treat each answer as a document and sample the topic assignments on it. Each sentence in the answer will get a topic that is consistent with the topic its words are assigned. This leads to a "bag of sentence topic" representation for multi-sentence answers, which can capture high level information rather than individual words. In our experiment, the sentences are segmented with Chinese punctuations (including comma), and the number of topic is set to 200.

3.3 Tensor Relevance Model

To model the relevance of question-answer pairs, we use a three-way tensor to transform the representations of question and answer into a semantic matching representation, like Qiu and Huang (2015). The representations of a question and its related answer are separately mapped to the hidden layers with a nonlinear transformation:

$$h_q = \sigma(W_h \cdot q + b_h) \tag{1}$$

$$h_a = \sigma(W_a \cdot a + b_a) \tag{2}$$

A new hidden layer h_{tensor} is added to model the interaction between question and answer via a three-way tensor W_{tensor}:

$$h_{tensor} = h_a W_{tensor}(h_q^T) \tag{3}$$

where T denotes tensor transformation.

We then use a logistic regression layer to calculate the final score:

$$score_{imp} = \sigma(W_{output} \cdot h_{tensor} + b_{output}) \tag{4}$$

where W_{output} and b_{output} are parameters of the regression layer.

4 The Extended Model ExMaLSTM

In cQA, the most appropriate answer to a question often explicitly mentions some key points of the question. However, as discussed above, we may lose this detailed matching information when embedding texts into the latent semantic space by using traditional neural network models. To overcome this limitation, we extend the basic LSTM neural network model (as shown in Figure 1) to incorporate the explicit matching knowledge, and then calculate the question-answer relevance by combining both implicit relevance score and explicit matching score in a dynamic fashion. Our extended model ExMaLSTM is depicted in Figure 2, where the notation 1, 2 and 3 are related to the following subsections 4.1, 4.2 and 4.3, respectively.

4.1 Extracting Explicit Matching Features

For each word in the question, we introduce the following explicit lexical matching features. These features describe how well each question word is explicitly matched in the answer as a possible key point.

In traditional lexical matching methods, only exact word matching features are used. In this paper, we extract explicit matching information from nine dimensions, by using external resources like synonym dictionary and word vectors pre-trained on a large corpus. So our explicit matching features have stronger power to capture the matching information in question-answer pairs.

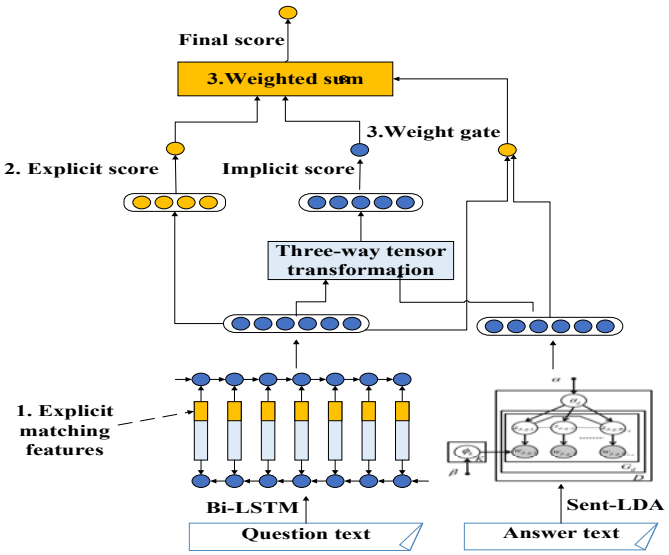

Fig. 2. Our extended ExMaLSTM model. The yellow parts denote our extensions with explicit matching knowledge, compared with the basic LSTM model in Figure 1.

- **word occurrence**. A boolean feature denoting whether the word occurs in the answer.
- **word occurrence count.** The number of occurrences of the word occurring in the answer.
- **synonym occurrence**. A boolean feature denoting whether any synonym of the word occurs in the answer. We use HIT-CIR Tongyici Cilin as our synonym dictionary.
- **synonym occurrence count**. The number of occurrences of synonyms occurring in the answer.
- **occurrence in the head**. A boolean feature denoting whether the word or its synonym occurs in the first sentence of the answer.
- **word2vec similarity**. The similarity score of the most similar word in the answer, which is calculated by cosine similarity between word vectors.
- **tf-idf score**. The tf-idf score of the word if any synonym or the word itself occurs in the answer.
- **content word**. A boolean feature denoting whether the word that occurs in the answer is a content word.
- **entity word**. A boolean feature denoting whether the word that occurs in the answer is an entity word (with POS tag NR, NT or NS).

4.2 Calculating Explicit Matching Score

For each word in the question, we form the new input representation of the LSTM layer by appending these lexical matching features to the original word vector. Then a bi-directional LSTM network is used to generate the semantic presentation h_q of a question text. By summarizing both the distributed semantic representation and the matching features of each word, the output of the LSTM layer h_q now captures two aspects of information:

1) The latent semantics of the question itself;
2) The overall status about how well the key points of the question are matched explicitly in the answer.

Then the explicit matching score is calculated by adding a nonlinear transformation layer on the hidden representation h_q, and then a logistic layer is employed to get the final output:

$$score_{exp} = \sigma(W_{exp} \cdot L(h_q) + b_{exp}) \tag{5}$$

where L denotes the nonlinear transformation layer, W_{exp} and b_{exp} denote the parameters of the regression layer.

In our model, the matching features are processed sequentially, thus the consecutively matched substrings can be extracted just like the Maximum Common Substring (MCS) method. Then they are treated as a whole to estimate the matching score. For example in Table 2, the Chinese word "spicy hot pot" is wrongly segmented into three words "spicy", "hot" and "pot", but our model can still capture the explicit matching information of this word by processing the consecutively matched fragments as a whole. So our model is robust to Chinese word segmentation errors, which is a non-trivial task for Chinese language processing in web texts like cQA.

Table 2. An example of a question-answer pair. The wrongly segmented fragments [spicy-hot-pot] are extracted by our model as a whole to estimate the matching score.

Question: I want to know where is the most delicious [**spicy-hot-pot**] in Beijing?
Expected Answer: On a cold winter day, you may like to have something hot with your family. Then the [**spicy-hot-pot**] is perhaps the best choice for you. Now let's introduce the most famous [**hot-pots**] in Beijing below ...

4.3 Combining Implicit and Explicit Scores

We calculate the final relevance score by combing both the implicit relevance score and the explicit matching score to take advantage of both aspects of information.

$$score = g.score_{imp} + (1-g).score_{exp} \tag{6}$$

Instead of using a fixed weight factor, we propose to utilize a dynamic scoring strategy that can dynamically assign appropriate weights to two relevance scores:

$$g = \sigma(W_{sel} \cdot [h_q, h_a] + b_{sel}) \tag{7}$$

Here, g is calculated via a gate network, which dynamically estimates the importance of two relevance scores based on the current hidden states. It avoids over-estimating or under-estimating the final score due to the arbitrary weight setting, since the question and answer with few matching words can be highly relevant, and vice versa, the question and answer with many common words may talk about different things.

5 Experiment

5.1 Experiment setup

The dataset comes from Baidu Zhidao, which is one of the most popular cQA services in China. We have crawled 180,000 question-answer pairs under the "travelling" topic. The data was pre-processed with Chinese word segmentation and part of speech tagging using ICTCLAS (Zhang et al., 2003). We also trained a CRF-based entity recognizer to annotate the places with the label NS. We removed the pairs which have too short answer or question (<=5 words) or consist of only an URL string. Finally, there are 160,000 question-answer pairs remained. We picked 5,000 pairs for testing, 5,000 for validation, and the remaining 150,000 for training.

In our experiment, the word embeddings are pretrained using word2vec (Mikolov et al., 2013) on Baidu Zhidao corpus, including the whole data of 160,000 question-answer pairs. The dimension is set to 200. The hyper-parameters in the neural network are tuned using the validation data, and Table 3 shows these settings. We employ a large margin objective for model training, and the objective loss function is optimized using AdaGrad.

Table 3. Hyper-parameters in the neural network

Hyper-parameters	Value
h_q length	200
h_a length	200
tensor rank	3
tensor output length	100
margin	0.2
λ of l2-norm	0.001

5.2 Baselines

We conduct extensive experiments on the dataset, including traditional bag-of-words methods and various neural network models.

- **Cosine similarity**. Calculate cosine similarity between vectors of question and answer using tf-idf weight.

- **KL divergence**. Construct the unigram language model M_q for a question and M_a for an answer, and then compute their KL-distance.
- **Word overlapping**. Simply count the number of overlapped words between question and answer.
- **DeepMatch**. We implement the DeepMatch network proposed by Lu and Li (2013). The number of latent topic is tuned to 200.
- **Bi-CNN models.** Both question and answer are embeded into the latent semantic space using convolutional neural network. This includes the Arc-1 model with multi-layer perception (Hu et al., 2014) denoted as "qCNN-aCNN-Mlp" and tensor model (Qiu and Huang, 2015) denoted as "qCNN-aCNN-Tensor".
- **One side CNN models**. We replace the answer side CNN in the above models with the Sent-LDA method discussed in Section 3.2, thus form two models denoted as "qCNN-aTopic-Mlp" and "qCNN-aTopic-Tensor", respectively.
- **One side LSTM models**. We use the LSTM network on question side and the Sent-LDA topic representation on the answer side. The "qLSTM-aTopic-Cosine" model calculates the cosine similarity between hidden representations like Palangi et al. (2015), while our basic model in this paper "qLSTM-aTopic-Tensor" utilizes a tensor layer.
- **LSTM model + Cosine**. We combine the basic LSTM network score and cosine similarity score in a straightforward way $score = a \cdot score_{lstm} + (1-a) \cdot score_{cosine}$, where the heuristic weight a is fine-tuned in the validation data.

5.3 Results

Table 4 reports the experimental results on answer retrieval in our cQA dataset. In general, the performances of traditional methods, including cosine similarity, KL-divergence and word overlapping are unsatisfying. However, we can still see that the simple exact word matching method like "Word over-lapping" retrieves correctly 32.3% answers in the 5,000 test data. It demonstrates that explicit lexical matching features play an important role for answer retrieval.

We get the following observations from Table 4. 1) The neural network models obtain considerably better results than traditional methods. 2) The tensor network gives an obvious improvement than the multi-layer perception. 3) Our Sent-LDA representation for answers obtains comparable results with CNN. 4) The bi-directional LSTM representation for questions achieves further improvement than CNN. 5) Among those neural network models with only implicit semantic relevance, the basic LSTM model in this paper performs the best with 46.8% on P@1 and 64.9% on MRR.

Table 4. The experimental results on answer retrieval. We implement various neural network models in our dataset: DeepMatch (Lu and li, 2013), qCNN-aCNN-Mlp (Hu et al., 2014), qCNN-aCNN-Tensor (Qiu and Huang, 2015), qLSTM-aTopic-Cosine (like Palangi et al., 2015).

Method	P@1	MRR
Cosine similarity	34.2	53.8
KL-divergence	24.4	45.1
Word overlapping	32.3	52.0
DeepMatch	41.9	60.5
qCNN-aCNN-Mlp	43.5	61.7
qCNN-aCNN-Tensor	44.6	63.8
qCNN-aTopic-Mlp	43.2	62.5
qCNN-aTopic-Tensor	45.4	63.2
qLSTM-aTopic-Cosine	45.7	62.7
qLSTM-aTopic-Tensor	**46.8**	**64.9**
+Cosine	47.7	65.8
+ Explicit match (static)	48.4	66.3
+Explicit match (dynamic)	**49.1**	**66.9**

Our proposed ExMaLSTM model combines dynamically the implicit semantic relevance score and the explicit matching information, achieving the best performance with 2.3% increase on P@1 ($p<0.01$) and 2.0% increase on MRR ($p<0.01$).

5.4 Analysis

We will give a more detailed analysis on our extended model ExMaLSTM. The incorporation of the explicit matching features is denoted as "Explicit match" in Table 4. For the static version, we simply add up the explicit and implicit scores with a fixed weight, which is fine-tuned on the validation data. For the dynamic version, we use the gate network to automatically calculate the weight.

It can be seen that the explicit matching information do benefit, because even simply combining the score of our basic LSTM network model with the cosine similarity score achieves better results than individual methods. Both of two versions consistently outperform the basic LSTM model. The static version obtains 1.6% increase on P@1 ($p<0.05$) and 1.4% increase on MRR ($p<0.05$), and the dynamic version obtains 2.3% ($p<0.05$) increase on P@1 and 2.0% increase on MRR ($p<0.05$). The dynamic weighting strategy performs better because it can consider different contributions of the explicit and implicit scores through the gated weight.

We briefly analyze the computation process on the example in Table 1. Table 5 gives the intermediate values in the relevance computing process between the question and its expected and unexpected answers.

Although the unexpected answer lies in the same "food" topic with the question thus gets a high implicit relevance score 0.91, the explicit matching score is quite low with only 0.22. What's more, the gate network does not give the implicit score a high weight (only 0.53). Thus the final relevance score between the question and its unexpected

answer is only 0.59. However for the expected answer, it gets both a high explicit score (0.87) due to the lexical matching of "spicy hot pot" in this question-answer pair, and a high implicit score (0.64) by addressing the same "food" topic. Thus the final relevance score between the question and its expected answer is 0.77. In this way, our extended model ExMaLSTM successfully retrieves the expected answer but discards the unexpected answer.

Table 5. The intermediate values in the relevance computing process.

Value	Expected Ans.	Unexpected Ans.
$score_{imp}$	0.64	0.91
$score_{exp}$	0.87	0.22
g	0.42	0.53
$score$	0.77	0.59

6 Conclusion

In this paper, we propose an ExMaLSTM model to incorporate the explicit matching knowledge into the traditional LSTM neural network. First, we extract the explicit lexical matching knowledge by using rich linguistic information. Then, we incorporate these explicit matching features into the LSTM network to summarize the overall explicit matching status between a question and its related answers. Finally, we dynamically assign different weights to the explicit matching score and the implicit relevance score through a gate network, and sum up both scores to get the final relevance score. The experimental results show that our proposed ExMaLSTM model outperforms various state of-the-art neural network methods.

Acknowledgement.
This work is supported by the National High Technology Research and Development Program of China (2015AA015403), the National Natural Science Foundation of China (61371129).

References

1. Ahmed Kishky, Song Yanglei,Wang Chi, Voss Clare, and Han Jiawei. 2014. Scalable topical phrase mining from text corpora. In Proceedings of the VLDB Endowment, pages 305–316.
2. Alex Graves, Abdel rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6645–6649.
3. Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. Advances in Neural Information Processing Systems (NIPS), pages 2042–2050.

4. Baoxun Wang, Bingquan Liu, Xiaolong Wang, andChengjie Sun andDeyuan Zhang. 2011. Deep learning approaches to semantic relevance modeling for chinese question-answer pairs. ACM Transactions on Asian Language Information Processing (TALIP), 10.
5. Chris Dyer, Miguel Ballesteros, Wang Ling, AustinMatthews, and Noah A. Smith. 2015. Transitionbased dependency parsing with stack long short-term memory. In Proceedings of ACL, pages 334–343.
6. Di Wang and Eric Nyberg. 2015. A long short-term memory model for answer sentence selection in question answering. In Proceedings of ACL, pages 707–712.
7. Haifeng Hu, Bingquan Liu, BaoxunWang, Ming Liu, and Xiaolong Wang. 2013. Multimodal dbn for predicting high-quality answers in cqa portals. In Proceedings of ACL, pages 843–847.
8. Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2015. Deep sentence embedding using the long short term memory network: Analysis and application to information retrieval. IEEE/ACM Transactions on Audio, Speech, and Language Processing, pages 694–707.
9. He, Jianshu Chen, Xinying Song, and Rabab Ward. 2015. Deep sentence embedding using the long short term memory network: Analysis and application to information retrieval. IEEE/ACM Transactions on Audio, Speech, and Language Processing, pages 694–707.
10. Hua-Ping Zhang, Hong-Kui Yu, De-Yi Xiong, and Qun Liu. 2003. Hhmm-based chinese lexical analyzerictclas. In SIGHAN '03 Proceedings of the second SIGHAN workshop on Chinese language processing, volume 17, pages 184–187.
11. Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. Proceedings of AAAI.
12. Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daum´e III. 2014. A neural network for factoid question answering over paragraphs. In Proceedings of EMNLP, pages 633–644.
13. Sepp Hochreiter and J¨urgen Schmidhuber. 1997. Long short term memory. Neural Comput, 9(8):1735–1780.
14. Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. 2016. A deep architecture for semantic matching with multiple positional sentence representations. Proceedings of AAAI.
15. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. Workshop at ICLR.
16. Wenpeng Yin and Hinrich Sch¨utze. 2015. Multigrancnn: An architecture for general matching of text chunks on multiple levels of granularity. In Proceedings of ACL, pages 63–73.
17. Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. In Proceedings of ACL, pages 1744–1753.
18. Xiaoqiang Zhou, Baotian Hu, Qingcai Chen, Buzhou Tang, and Xiaolong Wang. 2015. Answer sequence learning with neural networks for answer selection in community question answering. In Proceedings of ACL, pages 713–718.
19. Xipeng Qiu and Xuanjing Huang. 2015. Convolutionalneural tensor network architecture for communitybased question answering. Proceedings of IJCAI, pages 1305–C1311.
20. Zhengdong Lu and Hang Li. 2013. A deep architecture for matching short texts. Advances in Neural Information Processing Systems (NIPS), pages 1367–1375.