Incorporating Statistical Features in Convolutional Neural Networks for Question Answering with Financial Data

Shijia E
Tongji University
Shanghai, China
436_eshijia@tongji.edu.cn

Shiyao Xu
Tongji University
Shanghai, China
1452221@tongji.edu.cn

Yang Xiang
Tongji University
Shanghai, China
shxiangyang@tongji.edu.cn

ABSTRACT

The goal of question answering with financial data is selecting sentences as answers from the given documents for a question. The core of the task is computing the similarity score between the question and answer pairs. In this paper, we incorporate statistical features such as the term frequency-inverse document frequency (TF-IDF) and the word overlap in convolutional neural networks to learn optimal vector representations of question-answering pairs. The proposed model does not depend on any external resources and can be easily extended to other domains. Our experiments show that the TF-IDF and the word overlap features can improve the performance of basic neural network models. Also, with our experimental results, we can prove that models based on the margin loss training achieve better performance than the traditional classification models. When the number of candidate answers for each question is 500, our proposed model can achieve 0.622 in Top-1 accuracy (Top-1), 0.654 in mean average precision (MAP), 0.767 in normalized discounted cumulative gain (NDCG), and 0.701 in bilingual evaluation understudy (BLEU). If the number of candidate answers is 30, all the values of the evaluation metrics can reach more than 90%.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Neural networks**; *Learning latent representations*;

KEYWORDS

question answering; convolutional neural network; representation learning

ACM Reference Format:

Shijia E, Shiyao Xu, and Yang Xiang. 2018. Incorporating Statistical Features in Convolutional Neural Networks for Question Answering with Financial Data. In *WWW '18 Companion: The 2018 Web Conference Companion, April 23–27, 2018, Lyon, France*. ACM, New York, NY, USA, 5 pages. <https://doi.org/10.1145/3184558.3191826>

1 INTRODUCTION

Question answering (QA) is a critical task in the research areas of natural language processing (NLP) and information retrieval (IR). Retrieval-based QA systems [1] are mainly focused on measuring

the semantic similarity of two sentences and selecting the possible answer to the question by ranking one or more sentences [15] [14]. Therefore, the question answering task with financial data can be defined as that given a question q and an answer candidate pool $\{a_1, a_2, \dots, a_n\}$; we should select the best answer a_i , where $1 \leq i \leq n$. Due to the short text characteristic of the questions and answers, the current text matching methods [2] cannot be directly used in this task. In this paper, we primarily pay attention to the following problems:

1. *How to get the most valuable features of the QA pairs?* After analyzing the financial dataset, we find the QA pairs mostly are short texts. The length limitation of QA pairs leads to the sparsity of word occurrence, which is usually one or two. This feature implies that every word may have an important semantic information.

2. *How to train a model that can rank the positive answers before the negative ones in QA dataset?* As for the financial dataset, the positive answer to a question must be different from the negative one to some degree. In the task, we put the positive and the negative answers into the answer candidate pool for every question.

As a result, we tackle the matching problem mainly by convolutional neural networks (CNN) [8], and further improve the performance by incorporating the specific features. More precisely, we preprocess the financial dataset with TF-IDF and word-overlap to generate extra input features for the QA texts. After that, we use the proposed QA framework to extract the important features which can produce the distributed representations of questions and answers with neural layers.

The rest of this paper is structured as follows. Section 2 contains related work. In Section 3, we describe the definitions and implementations of the preprocessing and analyze the model we use. Experimental results and discussions are presented in Section 4, and finally, we give some concluding remarks in Section 5.

2 RELATED WORK

As mentioned above, the goal of the answer selection task is to find the best candidate answer. If the selected answer is contained in the ground truth set of the corresponding question, the prediction result is considered to be correct. Otherwise, it is incorrect. For this reason, the task can be treated as a binary classification problem. In addition to the distributed representation of questions and answers, another important thing is to give a metric to measure the matching degree of the QA pairs. Feng et al. [4] present a framework based on CNN. The questions and answers share the same CNN layers to represent the features. It also attempts several general similarities metrics such as cosine similarity. Similarly, Pang et al. [5, 9, 10] propose the models based on CNN for matching natural language sentences. Tan et al. [13] consider the shortcomings of CNN and

This paper is published under the Creative Commons Attribution 4.0 International (CC BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.

WWW '18 Companion, April 23–27, 2018, Lyon, France

© 2018 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC BY 4.0 License.

ACM ISBN 978-1-4503-5640-4/18/04.

<https://doi.org/10.1145/3184558.3191826>

adopt the long short-term memory network (LSTM) to model the QA pairs. LSTM is essentially a recurrent neural network (RNN), and the learned features can retain the word order, so as to further improve the overall performance of the model.

3 MODEL DESCRIPTION

Our question answering model (QA-Model) is based on CNN. The CNN has achieved good performance in the domain of image processing [7, 8] and machine translation [3]. We can leverage the experience of applying CNN to images and then fit it to the financial QA task. In a CNN, the kernel size is always much smaller than the input size. As a consequence, the output is only interactive with a narrow window of the input. Another feature of CNN is that it reuses the kernel parameters in the convolution operations, while the element in the weight matrix of traditional neural networks will be used only once to calculate the output. Also, with the k -MaxPooling combined to a CNN, we can always have equivariant representation. As the overall framework shown in Figure 1, the question word sequence is in the input layer, and the embedding layer takes it as the input to produce the question's vector representation for the next convolution layer. In what follows, the output of the convolution layer is the question's feature representation. The following are the max pooling layer and activation layer. In addition, the processing pipeline of the answer is same as the question. The final result of the input QA pair is the matching score between the two vector representations. The following is a detailed description of the proposed QA-model.

3.1 Embedding Layer

As shown in Figure 1, the input of the CNN is a sequence of words ($\mathbf{q} = \langle q_1, q_2, \dots, q_d \rangle$, d is the length) and the word vector ($\mathbf{V}$, $\mathbf{V} \in \mathbf{R}^d$) dimension is m . All word vector representations are drawn from the word embedding ($\mathbf{W} \in \mathbf{R}^{|V| \times d}$) which is trained by *word2vec* [12] based on the financial training data.

3.2 Convolution Layer

Convolution layer is a special linear computation, used for extracting patterns. Let the input sequence be $\mathbf{q} = \langle q_1, q_2, \dots, q_d \rangle$ (d is the fixed length) and i -th word's vector is $\mathbf{V}_{qi} = \langle v_1, v_2, \dots, v_m \rangle$ (m is the length of the embedding vector) which is derived from the matrix $\mathbf{R}^d$. The convolution layer projects the input layer into the feature maps, and the computation follows:

$$\mathbf{P} = \text{ReLU}(\mathbf{W} * \mathbf{V}_q + \mathbf{b}) \quad (1)$$

where convolution weight $\mathbf{W}$ and the vector $\mathbf{b}$ are the parameters that need to be learned. *ReLU* is the activation function (shown as the activation layer in Figure 1) which is widely used in CNN. $\mathbf{P}$ is the generated representation of the input. Besides, each convolution layer's stride is decided by users. As for this task, because the length of the sentence is short, we use one as the stride, so as to maximize the acquisition of multiple semantic representations.

3.3 Pooling Layer

To capture the important information of QA pairs, we use pooling function to combine the all representations $\mathbf{P}$ extracted from the previous convolution layers. In this paper, we prefer 1-max pooling,

and it means we consider the strongest values that represent the most important feature in one filter.

3.4 Additional features

3.4.1 TF-IDF feature. Term frequency-inverse document frequency (TF-IDF) [11] can be used to determine what words in a corpus of documents might be favorable to use in a sentence. Considering the financial QA dataset, almost every topic of the QA pair is different from the others. It means that the keywords of every QA pair differ from the others. For every QA pair, the **TF** of the keyword is lower than the **TF** of the remaining words. Equally, the **IDF** of keywords is much higher than the remaining words. **IDF** is computed as follows:

$$\text{IDF} = \log \frac{|D|}{1 + |j : t_i \in d_j|} \quad (2)$$

where $|D|$ is the number of all QA pairs, and $|j : t_i \in d_j|$ denotes the number of documents containing the word t_i . Therefore, we will compute TF-IDF ($\text{TF} \times \text{IDF}$) values of all words, and those values will be used to build our proposed models.

3.4.2 Word overlap feature. The word overlap (WO) is computed as follows:

$$\text{WO} = \frac{n_{\text{common}}}{n_q} \quad (3)$$

where n_{common} is the number of words that are located in both the question and the answer, and we can call that *common words*. n_q is the number of words in the question. The WO feature and common words will be used to enhance the representation of QA pairs.

3.4.3 Combination of the additional features and neural networks. With the features of TF-IDF and word overlap, we propose following feature combination methods.

TF-IDF embedding. For a word, we concatenate the original word vector and the TF-IDF value of that word to generate the TF-IDF embedding.

Simple overlap. We concatenate the cosine matching score and the WO value of the QA pair, and feed this representation to the fully connected neural layers. The output of the last layer is the final score, and it will be used to judge the matching degree of the QA pair.

TF-IDF overlap. In this method, we use all the TF-IDF values of the common words. The TF-IDF overlap is computed as follows:

$$\text{TF-IDF overlap} = \frac{\sum_{n=0}^N (\text{TF-IDF})_n}{n_q} \quad (4)$$

where N is the number of common words with the QA pair.

3.5 Training loss function

During the training process, we always prepare a positive answer (A_+) and a negative one (A_-) for each question (Q). With the convolution layers and pooling layers, our QA-Model can capture the distributed representations for the question (V_Q) and the answers (V_{A_+} and V_{A_-}). At last, we use cosine similarity to compute the matching score of QA pairs including $\text{cosine}(V_Q, V_{A_+})$ and $\text{cosine}(V_Q, V_{A_-})$.

Figure 1: The overall architecture of our proposed solution.

As a result, we minimize a ranking objective function defined as follows:

$$L = \max(0, M - \cos(V_Q, V_{A+}) + \cos(V_Q, V_{A-})) \quad (5)$$

where M is the constant margin, and L is the ranking loss value. Obviously, the loss function is aimed to enlarge the distance between the positive answer and the negative one. If $\cos(V_Q, V_{A+}) > \cos(V_Q, V_{A-}) + M$, it means that the training process makes the positive answer more suitable for the question than the negative one. Otherwise, if $\cos(V_Q, V_{A+}) - \cos(V_Q, V_{A-}) < M$, the neural network needs to update the parameters. In the test phase, we utilize the QA-Model to get the representations of QA pairs and treat the answer with the largest matching score as the correct candidate answer for the corresponding question.

4 EXPERIMENTS

4.1 Dataset

We apply our proposed models to this QA task with financial data. In this dataset, each QA pair has a label manually annotated with 1 or 0, where 1 means the answer is positive to the question, and 0 means the one is negative. There are 17,110 positive QA pairs in the training data which contain 6,648 questions and 57,638 answers.

4.2 Experimental Setup

4.2.1 Parameter settings. Before the training process, all the questions and answers were pre-processed, such that the sentence was segmented to a word sequence, where conjunctions, exclamations, and punctuation marks were removed. We tried several different margin values, such as 0.009, 0.1 and 0.5, and finally, we set M as 0.009 according to the experimental results on the training dataset based on the result of 10-fold cross-validation. During the training process, we utilized the pre-trained 100-dimensional word embedding as the initial embedding matrix. We trained our model with Adam [6] as the final optimization strategy. The batch size was 256. We mixed multiple convolution filters with length 2, 3, 5, and 7. All of the hyper-parameters were tuned based on measuring the cross-validation results.

4.2.2 Baseline. To further evaluate our models, we also set up a baseline model. We directly concatenate the distributed representations of the question and answer shown in Figure 1 as the feature vector, so that we can use a classification model to output the label of the QA pair. Therefore, 1 means the positive QA pair, and 0 means the negative QA pair.

Table 1: The experimental results of the 10-fold cross-validation with 500 candidate answers for each question.

Model type	Additional feature	#Filter	Top-1@500	MAP@500	NDCG@500	BLEU@500
Classification	-	400	0.125	0.192	0.392	0.178
	TF-IDF embedding (f1)		0.132	0.200	0.399	0.186
	Simple overlap (f2)		0.304	0.322	0.506	0.344
	TF-IDF overlap (f3)		0.360	0.365	0.542	0.407
	f2 + f3		0.395	0.390	0.563	0.423
	f1 + f2 + f3		0.402	0.394	0.566	0.426
	f1 + f2 + f3	4000	0.412	0.408	0.581	0.443
Margin loss	-	400	0.418	0.445	0.608	0.472
	TF-IDF embedding (f1)		0.427	0.453	0.614	0.479
	Simple overlap (f2)		0.481	0.486	0.640	0.533
	TF-IDF overlap (f3)		0.563	0.552	0.682	0.610
	f2 + f3		0.563	0.551	0.691	0.604
	f1 + f2 + f3		0.575	0.561	0.699	0.614
	f1 + f2 + f3	4000	0.622	0.654	0.767	0.701

Table 2: The experimental results of the 10-fold cross-validation with 30 candidate answers for each question.

Model type	Additional feature	#Filter	Top-1@30	MAP@30	NDCG@30	BLEU@30
Classification	f1 + f2 + f3	400	0.736	0.753	0.839	0.738
		4000	0.783	0.802	0.871	0.752
Margin loss	f1 + f2 + f3	400	0.867	0.892	0.930	0.874
		4000	0.928	0.942	0.963	0.930

4.3 Results and Discussions

The performance of our proposed models is evaluated by Top-1 accuracy, MAP, NDCG, and BLEU defined by this task. Table 1 and Table 2 show the experimental results of the 10-fold cross-validation in detail.

We prepare 500 candidate answers for each question to evaluate the performance of different models shown in Table 1. We can find that the results obtained by margin loss training are better than those of the traditional classification model, no matter what features are used. Moreover, each of the proposed additional features, especially the TF-IDF overlap, can effectively improve the performance. For the NDCG metric, when we adopt the margin loss training with 400 convolution filters, the use of all additional features can yield nearly 10% absolute improvement over the results without any additional features. We also find that the number of convolution filters has a great influence on the performance of the models. The model with 4000 convolution filters is better than the 400 convolution filters under the same condition. In Table 2, we show the performance with 30 candidate answers. We can find that all the evaluation metrics can reach more than 90% in this case.

5 CONCLUSION

In this paper, we study the QA task with financial data by applying the CNN based deep learning models. Furthermore, the proposed models can perform better by incorporating the statistical features. The solution is an end-to-end framework which does not need manual intervention, so it can also be easily migrated to other similar tasks in the future.

ACKNOWLEDGMENTS

The work is supported by the National Basic Research Program of China under Grant No.: 2014CB340404, National Natural Science Foundation of China under Grant No.: 71571136 and Science and Technology Commission of Shanghai Municipality under Grant No.: 16JC1403000, 14511108002.

REFERENCES

- [1] Li Cai, Guangyou Zhou, Kang Liu, and Jun Zhao. 2011. Learning the Latent Topics for Question Retrieval in Community QA. In *IJCNLP*, Vol. 11. 273–281.
- [2] Asli Celikyilmaz, Dilek Hakkani-Tur, and Gokhan Tur. 2010. LDA based similarity modeling for question answering. In *Proceedings of the NAACL HLT 2010 Workshop on Semantic Search*. Association for Computational Linguistics, 1–9.
- [3] Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard M Schwartz, and John Makhoul. 2014. Fast and Robust Neural Network Joint Models for Statistical Machine Translation. In *ACL (1)*. 1370–1380.
- [4] Minwei Feng, Bing Xiang, Michael R Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. In *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*. IEEE, 813–820.
- [5] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*. 2042–2050.
- [6] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
- [7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.
- [8] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.
- [9] Zhengdong Lu and Hang Li. 2013. A deep architecture for matching short texts. In *Advances in Neural Information Processing Systems*. 1367–1375.
- [10] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text Matching as Image Recognition. In *AAAI*. 2793–2799.

- [11] Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, Vol. 242. 133–142.
- [12] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. <http://is.muni.cz/publication/884893/en>.
- [13] Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108* (2015).
- [14] Haocheng Wu, Wei Wu, Ming Zhou, Enhong Chen, Lei Duan, and Heung-Yeung Shum. 2014. Improving search relevance for short queries in community question answering. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 43–52.
- [15] Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. 2014. Deep learning for answer sentence selection. *arXiv preprint arXiv:1412.1632* (2014).