Question Similarity Modeling with Bidirectional Long Short-Term Memory Neural Network

Chao An doitfoolac@gmail.com
College of Computer, NUDT
Jiuming Huang jiuming.huang@qq.com
College of Computer, NUDT
Shoufeng Chang changsf626@hotmail.com
Beijing Satellite Navigation Center
Zhijie Huang hzj@gaodig.com
Beijing Gaodi Information Technology Co., Ltd.

***Abstract*—Modeling sentence similarity all along is a challengeable task in the field of natural language processing (NLP), since ambiguity and variability of linguistic expression. Specifically, in the field of community question answering (CQA), homologous hotspot is focusing on question retrieval. To get the most similar question compared with user's query, we proposed a question model building with Bidirectional Long Short-Term Memory (BLSTM) neural networks, which as well can be used in other fields, such as sentence similarity computation, paraphrase detection, question answering and so on. We evaluated our model in labeled Yahoo! Answers data, and results show that our method achieves significant improvement over existing methods without using external resources, such as WordNet or parsers.**

Keywords—long short term memory; question retrieval; community question answering; sentence similarity

I. Introduction

With the explosive growth of the internet information, Community question answering (CQA) portals, such as Yahoo! Answers, Quora and Baidu Knows, are developing dramatically. Acting as a platform for people to share their knowledge and experience, CQA portals have accumulated a lot various forms of data from multiple fields organized as questions and a list of candidate answers. To get the intended answer in CQA, there are two steps. Firstly, retrieving similar posted questions and then gathering the candidate answers through those similar questions. Secondly, based on quality assessment on the candidate answers, picking up the most relevant answer. During the interactive process between users and the CQA website, sentence similarity computation, question similarity computation specifically, plays a key role. Particularly, TREC Live QA task is held for the related research.

TREC stands for Text REtrieval Conference, is a proceeding series of workshops centering on a list of different information retrieval (IR) research areas, or tracks. It is composed of several tasks, for example, Live QA task which is to generate answers to real questions coming from real users via a live question stream in real time. Our basic process for competition is intent understanding, similar questions retrieving, answer quality assessment and ranking. Since the mandatory real-time and casual form of questions, the question similarity computation plays a pivotal role.

However, owing to words ambiguity and diversity of sentence structure, measuring semantic relevance of two sentences is a very complicated task. Solving the fundamental issue will contribute a lot to many areas, such as paraphrase detection, question retrieval and question answering. To cope with this challenge, predecessors proposed a variety of methods, from logic-based inferencing methods to vector space semantic models, from traditional NLP syntactic parsing or lexical semantic networks based WordNet similarity measurement to recently fashionable deep neural networks, and these methods are gradually getting better performance.

In this paper, we propose a question similarity computation model adopting deep learning method. Our model architecture includes two components: 1) question modeling module. We model question using the Bidirectional Long Short Term Memory (BLSTM), one variant of the Recurrent Neural Network (RNN), to get trained distributed vector representation of question. 2) similarity measurement module. As for similarity measurement module, the previous works implemented several kinds of measurements, such as the simplest cosine similarity method, relatively complex supervised neural networks and so on. To focus on modeling question, we adopted fully connected neural networks uniformly as our similarity measurement module. Not only could our model be applied to question retrieval or question answering field, but most importantly, it is also significant to sentence similarity, paraphrase detection and paraphrase generation, etc.

II. Related Work

Not only previous work on modeling sentence similarity, but also recent popular methods are basically based on sentence features engineering, superficial or semantic level features or combination of several types of features. [1]Mihalcea et al. (2006) proposed vector based similarity method to represent sentence relevance via computing cosine similarity with text TF-IDF weighting. [2]Madnani et al. (2012) combined eight machine translation metrics to train paraphrase detection model. [3]Fellbaum (1998), [4]Fern and Stevenson (2008) employed

978-1-5090-1192-6/16 $31.00 © 2016 IEEE
DOI 10.1109/DSC.2016.13

knowledge-based method, using external lexical database (WordNet) to model sentence similarity. And [5]Hassan (2011) used distributional model, latent semantic analysis, mapping textual high-dimensional features to lower-dimensional latent semantic space for getting more abstract and concise representation of sentence meaning.

With the astonishing performance of Convolutional Neural Network (CNN) in image identification field, deep learning has become a hot research area and attracted quite a number of scholars to apply this technology to academic research or commercial applications. Recently, features engineering in NLP has moved from traditional artificial extraction to automatic extraction, and deep learning extracts features automatically very well. [6]Kalchbrenner et al. (2014) introduced a convolutional neural network to model sentence. [7]He et al. (2015) adopted multi-perspective convolutional neural networks and structured similarity layer to express sentence similarity.

Previous works showed that [8]Long Short Term Memory (LSTM) deep neural network are more suitable in modeling sequences, and [9]Mike Schuster and Kuldip K. Paliwal (1997) confirmed that bidirectional RNN has more advantages than unidirectional RNN in sentence modeling, because the previous can better represent the meaning of whole sentence. So we built question model with BLSTMs based on the comprehensive consideration of previous research results. To maximize question's semantic information utilization and questions' semantic distinction information collection, we design our model architecture based on analysis of multiple perspectives of input questions. We now go forward to describe our model in detail, and then display our model's performance evaluation.

III. Model Description

Sentence semantic understanding is almost an insurmountable peak in NLP, while statistical methods such as Artificial Neural Networks (ANNs) inferring the knowledge from data, open the door to the new world. With the development and evolution of the ANNs, different kinds of neural network models emerged to deal with different tasks. Multilayer perceptrons (MLPs), one type of ANNs, have the limitation to deal with time varying patterns, so Recurrent Neural Networks (RNNs) came into being which could tackle the issue. However, RNNs are not expected as in theory to handle long-term dependencies that will result in the gradient vanishing problem. As one variant of RNNs, Long Short-Term Memory Networks (LSTMs) are very special which not only avoids gradient vanishing but works much better than the standard version in many tasks. Next, we will introduce LSTMs at first, then give out our model architecture's detailed description.

A. Long Short-Term Memory Networks (LSTMs)

As a kind of RNNs, LSTMs are explicitly designed to avoid the long-term dependency problem. Different from the standard RNNs whose repeating module is just a single $\tanh$ layer, LSTMs has four layers, respectively input, input gate, forget gate and output gate, interacting in a very special way. Figure 1 and Figure 2 show the repeating module of RNNs and LSTMs respectively.

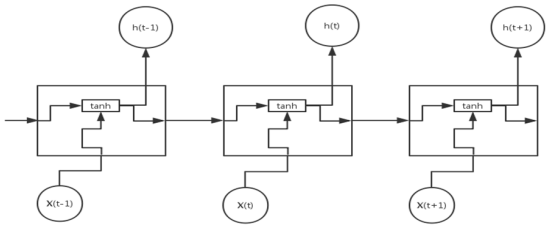

Figure 1: The repeating module in a standard RNN contains a single tanh layer.

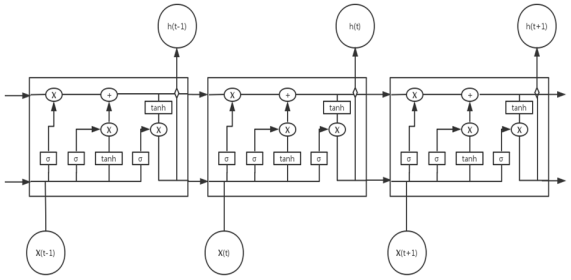

Figure 2: The repeating module in a LSTM contains four interacting layers. There are four gates as input, input gate, forget gate and output gate, and they are all composed out of a sigmoid neural net layer and a pointwise multiplication operation. The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. Based on this mechanisms, these gates can protect and control the cell state.

Here, we walk through the internal structure of LSTMs to explain how the LSTMs store and update its cell state in detailed as far as possible, and give related mathematical expression.

Firstly, As red marked part shown in Figure 3 I, the sigmoid layer acts as forget gate to decide what information should be forgot. Mathematical formula is expressed as follows.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \qquad (1)$$

Secondly, it turns to decide what new information we're going to store in the cell state, shown in Figure 3 II, highlighted in red. This includes two parts. On the one hand, a sigmoid layer called the "input gate layer" decides which values should be updated. On the other hand, a $\tan h$ layer creates a vector of new candidate values, that could be added to the state. In the next step, combine these two updated values to create an update to the state.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \qquad (2)$$

$$C'_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \qquad (3)$$

Next, update the cell state to the newest state on the basis of older state. See Figure 3 III, red part. The previous step has decided how to update cell state, and here we just need to multiply the old state by f_t , forgetting the things we decided to forget earlier. Then we add $i_t * C'_t$. This is the new candidate values, scaled by how much we decided to update each state value.

$$C_t = f_t * C_{t-1} + i_t * C'_t \quad (4)$$

At last, give the output. Figure 3 IV, marked in red represented. This output will be filtered by the sigmoid layer to decide what parts of the cell state we're going to output and the cell state will be normalize through $\tan h$, so that we only output the parts we wanted to.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

$$h_t = o_t * \tanh(C_t) \quad (6)$$

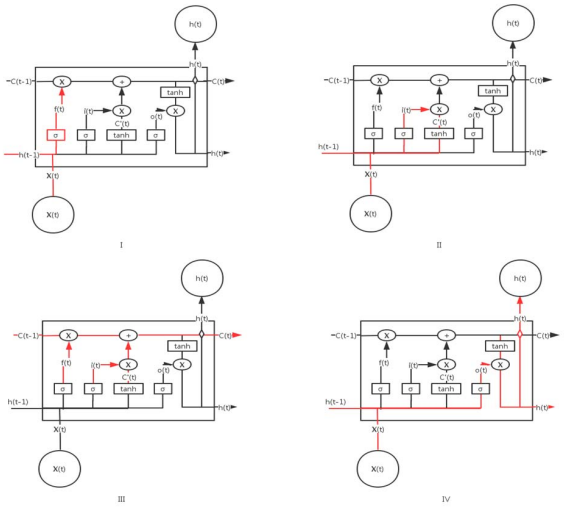

Figure 3: LSTM store and update its cell state

The above LSTM is a very normal variant, and there are a lot different types of LSTM, Gated Recurrent Unit (GRU) for example, but no details here.

B. Question modeling module

In this section we describe our bidirectional LSTM (BLSTM) network for modeling question. Bidirectional LSTM is a little different from unidirectional LSTM in architecture, see Figure 4. From the photo, we can know that BLSTM has a feedback part more than LSTM, considering back info also has an impact on the front in a sequential data, so BLSTM can better represent the whole meaning of a sentence or a question than LSTM.

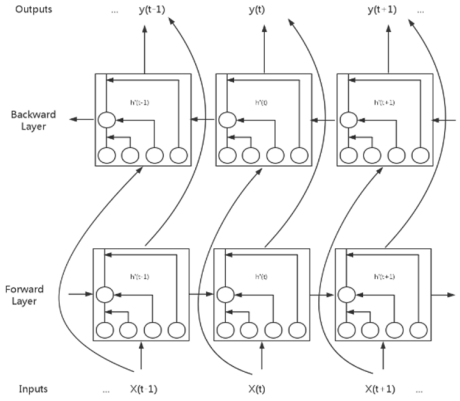

Figure 4: Bidirectional Long Short-Term Memory (BLSTM)

We built our question model using BLSTM, including two types. One is modeling each question in a question pair respectively using BLSTM, and the other is combining the two questions and using BLSTM to model them together. Figure 5 shows the two architectures.

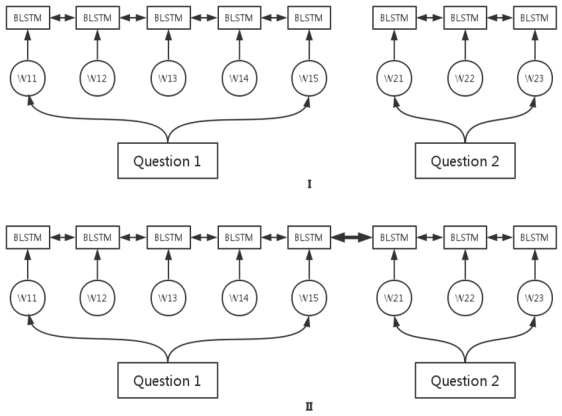

Figure 5: Two types of question modeling architecture. Question 1 and Question 2 stand for the two questions to be compared. {W11, W12, W13, W14, W15} is the result of segment of Question 1, taken as the input of BLSTMs. Question 2 takes the same operation. our BLSTMs takes word level input, and every BLSTM unit has the same internal structure as shown in Figure 4.

As viewed in Figure 5 , the only difference between I and II is the intermediate junction, bold double-headed arrow in black. Owing to comparing questions' similarity, so connecting two questions instead of isolating them, may be better to interpret the their relevance. And experimental results precisely confirmed our view.

Through the question modeling module, given input question, we can get the output which is the distributed vector representative of question. Based on the output, we can use the standard metrics like cosine similarity to get the question

relevance. Next, we will illustrate our similarity measurement module.

C. Similarity measurement module

Given two input questions, the first component of our model will output corresponding distributed vector. According to our previous introduction, the vector stands for the whole meaning of each question (Question modeling architecture I) or the whole meaning and relevance of two questions (Question modeling architecture II). In our experiments, we adopted fully connected layer uniformly as the similarity measurement layer. The fully connected layer uses sigmoid activation to get the probability distribution of classification result whose value ranges between 0 and 1. We tried to adjust the threshold for getting better performance, but the difference is not obvious, so we just take the default value 0.5 as our classification threshold.

IV. Experiments and Results

A. Task and Dataset

For quantizing the similarity of two questions, we built question similarity model to measure the relevance of two questions. About the dataset[1], it is referred by one paper in CIKM 2014 from Yahoo! Answers, authored by Wei Wu from Natural Language Computing Group of Microsoft Research Asia. We choose this dataset because it's corrected (basically includes several kinds of questions: How, What, Why, When, Where and so on) and labeled which can reduce a lot of human and financial resources. It is mainly composed of two files, one has just two questions per line and the other records the relation that question ID maps to several answers. In our experiments, we only use the previous file to compute questions' similarity without considering the answer text. However, in the future research, we will take the answer part into account.

B. Training

We use keras[2], one of popular deep learning frame to build our model. And the training objective is to minimize the Log Loss, just as the binary_crossentropy in keras available objectives does.

$$\text{logloss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij}) \qquad (7)$$

where N is the number of samples or instances, M is the number of possible labels, y_{ij} is a binary indicator of whether or not label j is the correct classification for instance i, and p_{ij} is the model probability of assigning label j to instance i. A perfect classifier would have a Log Loss of precisely zero. Less ideal classifiers have progressively larger values of Log Loss. If there are only two classes then the expression above can be simplified to formula (8)

$$\text{logloss} = -\frac{1}{N}\sum_{i=1}^{N}[y_i log p_i + (1-y_i)\log(1-p_i)]$$

Log Loss quantifies the accuracy of a classifier by penalizing false classifications. Minimizing the Log Loss is basically equivalent to maximizing the accuracy of the classifier. So during the training, we can see that the accuracy in validation data is gradually increased.

C. Experiment Settings

We conducted experiments using both one layer and two layers BLSTMs to model question, and tested on two question modeling architectures shown in Figure 5. In all, we executed four experiments using BLSTMs. We also carried out several other methods based on RNN, detailed parallel results are listed in section D. About the training of our model, we shuffled the dataset firstly for making the learning more effective. And we used 10-fold cross-validation to test our model's performance in the development dataset. Furthermore, to avoid the overfitting, we added a dropout layer in the end of our neural networks with 50% dropout ratio. In addition, for getting better model, we adopted adagrad optimizer to correct our model's parameters, set word vector length to 300 dimension and set the batch size to 100.

D. Results on Dataset

Based on the test dataset from Yahoo! Answer, we trained and saved our question similarity assessment model. Moreover, we recorded several experimental indexes, including accuracy, recall and F1-measure. Because of shuffling the dataset, each index will float on a small scale, so we averaged five experimental results for every index.

Model	Accuracy	Recall	F1
RNN	0.678	0.581	0.593
LSTM	0.713	0.661	0.657
BLSTM_I	0.692	0.651	0.639
BLSTM_II	**0.726**	**0.683**	**0.668**
2-layer BLSTM_I	0.697	0.652	0.644
2-layer BLSTM_II	0.698	0.658	0.639

Table 1: Test set results on the yahoo.data.dat referred by Wei Wu. Rows in grey are the approaches that models question using the architecture II in Figure 5.

E. Discussion and Conclusion

On the dataset, we tested several deep learning methods based on RNNs to compute question similarity. And about the question modeling, we adopted two kinds of architectures, connected and separated. The previous considers more correlation between two input questions than the latter and performs better, as the results shown in Table 1 that prove the point. As shown in Table 1, rows in grey are using the connected question modeling architecture, and compared with BLSTM_II, we can see that all the indexes, including accuracy, recall and F1-measure, the BLSTM_I all falls behind. Additionally, from the table, connected BLSTM is better than

[1] Dataset: https://onedrive.live.com/?id=8DE2D4C0D993976F%21277&cid=8DE2D4C0D993976F

[2] Keras: http://keras.io/

LSTM, and LSTM is better than RNN. However, more layers don't mean better performance given the results of BLSTM_II and 2-layer BLSTM_II, so deep neural networks do not necessarily have to be highly deep. In fact, deep learning is a new thing. During the tuning of model parameters, we tried several set of parameters, some are important to the experimental results, but some are not. So we need more in-depth investigation for better understanding and using the deep neural networks appropriately

In summary, we developed a basic model for question similarity based on recurrent neural network. It could be used for sentence similarity as well. And this model will be applied to TREC 2016 Live QA task in the near future. Last year, our team adopted a question similarity computation method based on LSA and syntactic parsing, and got the third place in all the teams. Using the new model, we can prospect for better results this year. Furthermore, future work could extend this model to related tasks including question answering, paraphrase detection, paraphrase generation and so on.

ACKNOWLEDGMENT

This work was supported in part by the National Key Fundamental Research and Development Program of China (2013CB329601), National Natural Science Foundation of China (61372191, 61202362, 61472433), Project funded by China Postdoctoral Science Foundation (2013M5452560, 2015T81129). Any opinions, findings, conclusions, or recommendations expressed are those of the authors and do not necessarily reflect the views of the sponsor. We would like to thank the anonymous reviewers for their feedback.

REFERENCES

[1] Mihalcea R, Corley C, Strapparava C. Corpus-based and knowledge-based measures of text semantic similarity[C]//AAAI. 2006, 6: 775-780.

[2] Madnani N, Tetreault J, Chodorow M. Re-examining machine translation metrics for paraphrase identification[C]//Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, 2012: 182-190.

[3] Fellbaum C. Towards a representation of idioms in WordNet[C]//Proceedings of the COLING/ACL Workshop on Usage of WordNet in Natural Language Processing Systems [online], Montreal. Available at: http://citeseer. ist. psu. edu/bhandari02machine. html.[Accessed March 2011]. 1998.

[4] Fernando S, Stevenson M. A semantic similarity approach to paraphrase detection[C]//Proceedings of the 11th Annual Research Colloquium of the UK Special Interest Group for Computational Linguistics. 2008: 45-52.

[5] Hassan S. Measuring semantic relatedness using salient encyclopedic concepts[M]. University of North Texas, 2011.

[6] Kalchbrenner N, Grefenstette E, Blunsom P. A convolutional neural network for modelling sentences[J]. arXiv preprint arXiv:1404.2188, 2014.

[7] He H, Gimpel K, Lin J. Multi-perspective sentence similarity modeling with convolutional neural networks[C]//Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. 2015: 1576-1586.

[8] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.

[9] Schuster M, Paliwal K K. Bidirectional recurrent neural networks[J]. Signal Processing, IEEE Transactions on, 1997, 45(11): 2673-2681.

[10] Jozefowicz R, Zaremba W, Sutskever I. An empirical exploration of recurrent network architectures[C]//Proceedings of the 32nd International Conference on Machine Learning (ICML-15). 2015: 2342-2350.

[11] Androutsopoulos I, Malakasiotis P. A survey of paraphrasing and textual entailment methods[J]. Journal of Artificial Intelligence Research, 2010: 135-187.

[12] Le Q V, Mikolov T. Distributed representations of sentences and documents[J]. arXiv preprint arXiv:1405.4053, 2014.

[13] Severyn A, Moschitti A. Learning to rank short text pairs with convolutional deep neural networks[C]//Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2015: 373-382.

[14] Zhang K, Wu W, Wu H, et al. Question retrieval with high quality answers in community question answering[C]//Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management. ACM, 2014: 371-380.

[15] Achananuparp P, Hu X, Shen X. The evaluation of sentence similarity measures[M]//Data warehousing and knowledge discovery. Springer Berlin Heidelberg, 2008: 305-316.

[16] Achananuparp P, Hu X, Zhou X, et al. Utilizing sentence similarity and question type similarity to response to similar questions in knowledge-sharing community[C]//Proceedings of QAWeb 2008 Workshop, Beijing, China. 2008, 214.

[17] Song W, Feng M, Gu N, et al. Question similarity calculation for FAQ answering[C]//Semantics, Knowledge and Grid, Third International Conference on. IEEE, 2007: 298-301.

[18] Gomaa W H, Fahmy A A. A survey of text similarity approaches[J]. International Journal of Computer Applications, 2013, 68(13).

[19] Jeon J, Croft W B, Lee J H. Finding similar questions in large question and answer archives[C]//Proceedings of the 14th ACM international conference on Information and knowledge management. ACM, 2005: 84-90.

[20] Iyyer M, Boyd-Graber J L, Claudino L M B, et al. A Neural Network for Factoid Question Answering over Paragraphs[C]//EMNLP. 2014: 633-644.

[21] Gupta P, Gupta V. A survey of text question answering techniques[J]. International Journal of Computer Applications, 2012, 53(4).

[22] Wu H, Wu W, Zhou M, et al. Improving search relevance for short queries in community question answering[C]//Proceedings of the 7th ACM international conference on Web search and data mining. ACM, 2014: 43-52.

[23] Bengio Y, Courville A, Vincent P. Representation learning: A review and new perspectives[J]. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2013, 35(8): 1798-1828.

[24] Goldberg Y. A Primer on Neural Network Models for Natural Language Processing[J]. arXiv preprint arXiv:1510.00726, 2015.

[25] Yin W, Schütze H. Convolutional neural network for paraphrase identification[C]//Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2015: 901-911.

[26] Wen T H, Gasic M, Mrksic N, et al. Semantically conditioned lstm-based natural language generation for spoken dialogue systems[J]. arXiv preprint arXiv:1508.01745, 2015.

[27] Yan X. Question Understanding and Similarity Computation Method Based on Semantic Analysis[C]//Proceedings of the 2012 International Conference on Information Technology and Software Engineering. Springer Berlin Heidelberg, 2013: 699-708.