# A Hybrid Neural Model for Identifying Similar Questions in Community Question Answering

Van-Tu Nguyen
Faculty of Natural Sciences and Technology, Tay Bac University, Son La, Vietnam
tuspttb@utb.edu.vn

***Abstract*—Community Question Answering (cQA) platforms have emerged as valuable resources for knowledge sharing, where users submit questions and obtain answers from the community. A central task in cQA is to automatically identify semantically similar questions and rank them according to their relevance to a new input query. In this paper, we propose a hybrid neural model that integrates Bidirectional Long Short-Term Memory (BLSTM) networks with a Multi-Layer Perceptron (MLP) to jointly learn question representations and compute similarity scores. To further improve similarity estimation, the model also incorporates auxiliary external knowledge features, such as question type and category, in a lightweight manner. We evaluate our approach on two benchmark datasets, SemEval 2016 Task 3 and Quora Question pairs. Experimental results show that our hybrid model consistently outperforms strong baselines and achieves competitive performance compared to recent state-of-the-art methods.**

***Keywords—community question answering, neural network, question similarity, external knowledge, BLSTM.***

## I. INTRODUCTION

Community Question Answering (cQA) forums archive millions of questions and answers created by users, providing a rich source of information that is often missing in web search engines and automatic QA systems. On these forums, users can freely submit questions and receive answers from other members of the community. Popular cQA platforms such as StackOverflow and Quora have become increasingly important in real-world applications. The questions and answers on these forums are highly diverse, enabling users to seek solutions from complex and heterogeneous information sources.

In cQA, when a user submits a new input question, if the question is similar to one that has already been answered (i.e., semantically similar), the system should be able to return the most relevant question–answer pairs and rank them according to their similarity to the input question.

Previous studies have addressed this problem by measuring the similarity between an input question and questions in the database. These measurements are typically based on various representations of input and related questions, such as standard features (e.g., n-grams) or linguistic information requiring deep analysis, such as syntactic parsing [1, 2, 3]. However, the choice of representations and features is often an empirical process, driven by intuition, experience, and domain expertise. Although using syntactic and semantic information has been shown to improve performance, it is computationally expensive and requires extensive external tools (e.g., syntactic parsers, lexicons, knowledge bases). Moreover, adapting such methods to new domains requires additional effort to tune feature extraction pipelines and integrate new resources, which may not even exist.

Recently, deep neural network–based learning methods have proven effective for Natural Language Processing (NLP) tasks such as semantic parsing [4], search query retrieval [5], sentence modeling [6], and sentence classification [7].

In this paper, we propose a hybrid model for determining the similarity between an input question and questions in the database. Our model uses Bidirectional Long Short-Term Memory (BLSTM) to jointly learn representations for input and related questions. The primary goal is to generate effective representations for computing similarity. To enhance performance, we also incorporate external knowledge in a lightweight manner.

We report experimental results on two cQA datasets: (1) SemEval 2016 Task 3, a large-scale non-factoid QA dataset from the Qatar Living forum, where our model demonstrates significant improvements compared to baseline methods; and (2) the Quora dataset, extracted from https://www.quora.com/, where our model outperforms several strong baselines.

The remainder of this paper is organized as follows. Section 2 describes related work on determining question similarity. Section 3 presents details of the proposed models. Sections 4 and 5 discuss experimental settings and results on the SemEval 2016 and Quora datasets, respectively. Finally, Section 6 concludes the paper.

## II. RELATED WORK

Identifying similar questions and ranking question–answer pairs related to an input query in cQA has become an essential task in designing effective cQA systems. A wide range of studies have explored methods for measuring similarity between input and related questions, or between a question and its answers, often using cosine similarity at the word level. In addition, some studies have proposed more advanced features and models. For example, Cao et al. [8] classified questions into different topics and used these features to build a recommendation system. Duan et al. [9] extracted question foci and employed them in similarity measurement. Other researchers [10, 11] adopted topic modeling approaches.

Zhou et al. [12] and Jeon et al. [13] followed a translation-based approach for question–answer pairs.

Several studies have relied on syntactic information. Wang et al. [3] used substructures of parsed trees as features to measure similarity between questions. The authors in [1, 2] also exploited syntactic information, employing tree kernel methods on parsed trees within the KeLP platform [14]. Franco-Salvador et al. [15] applied an SVM-rank method to distributed word representations for the ranking task at SemEval 2016 Task 3.

More recently, deep neural network–based methods have shown considerable promise in machine learning [16]. They have been particularly successful in image processing and speech recognition tasks, and are increasingly outperforming traditional sparse, linear models in NLP [6, 17]. Neural models have proven effective for sequence labeling [18], answer selection [19, 20], answer sentence selection [21], and ranking questions in cQA [22]. For example, dos Santos et al. [22] used CNNs and bag-of-words (BOW) representations of input and related questions to compute cosine similarity scores. Bahdanau et al. [23] introduced a neural attention model for machine translation, demonstrating that attention mechanisms can effectively handle long sentences. Mohtarami [24] proposed an LSTM- and BOW-based model to assess the relevance between questions and their answers.

Different from these previous studies, in this paper we propose a hybrid model for calculating similarity between input and related questions, and then using this similarity to rank question–answer pairs. Our model employs BLSTM to generate vector representations of input and candidate questions. It also incorporates additional information, such as question category, question type, and related answers, which helps improve performance compared with prior approaches.

## III. Our Approach

### *A. BLSTM-based Model (Baseline model)*

Bi-directional LSTM (BLSTM)

A BLSTM consists of two LSTM networks running in parallel: one processes the input sequence from left to right, and the other processes it from right to left. At each time step, the hidden state vector of the BLSTM is formed by concatenating the forward and backward hidden state vectors. This mechanism allows the model to capture both past and future contextual information. The output vector at each time step is therefore the concatenation of the two directional outputs, i.e., $\boldsymbol{h_t} = \overrightarrow{\boldsymbol{h_t}} || \overleftarrow{\boldsymbol{h_t}}$. Figure 1 illustrates the structure of a BLSTM.

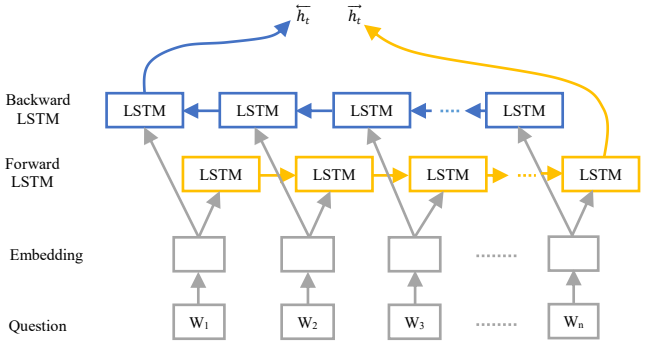


*Figure 1. Bidirectional LSTM*

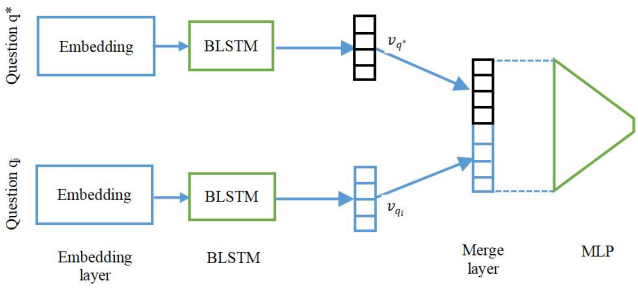


*Figure 2. The architecture of BLSTM-based model for calculating the similarity score between input question and related question*

We construct a BLSTM-based model to compute similarity scores for each input–related question pair, and then rank the related questions according to these scores. The overall architecture of the model is shown in Figure 2. First, the words of both input and related questions are converted into vectors using a pre-trained word2vec model. A tokenized question $q$ with $n$ words $\{w_1, w_2, \ldots, w_n\}$ is represented by word vectors $e_n \in R^d$, where $d$ is the dimension of the word vector. Combining all word vectors forms a question matrix embedding $E \in R^{n \times d}$, which is then fed into the BLSTM and processed in both directions. In this way, contextual information across words in both input and related questions is captured through the temporal recurrence of the BLSTM. The BLSTM produces vector representations for the input and related questions. A merge layer then concatenates these two vectors into a single representation, which is passed to a fully connected Multi-Layer Perceptron (MLP). The MLP consists of two hidden layers: the first is fully connected with the same number of neurons as the input vector size, while the output layer contains a single neuron that predicts the similarity score between the input and related question. This output neuron uses a sigmoid activation function to produce a probability value in the range [0, 1]. For training, we employ mean squared error as the loss function and optimize the model using Stochastic Gradient Descent (SGD). Model performance is evaluated using accuracy metrics collected during training.

### *B. External Knowledge*

The information obtained from each input–related question pair in the BLSTM-based model alone is not always sufficient to evaluate their similarity. To address this limitation, we incorporate external knowledge

features derived from the answers of related questions, question types, and question categories. These additional sources of information provide complementary signals for more accurate similarity estimation.

**Conventional Features**

We use some common features extracted from the surface forms of the question and answers, including: the ratio of the number of words between the input question and the related question; the ratio of the number of words between the input question and the answer of the related question; bag of word, word overlap, noun overlap, name entities overlap.

**Question Type**

In most cQA systems, input questions typically contain interrogative words that indicate their type. Question type is therefore a useful cue for identifying similarity between questions. To extract question type features, we define a set of interrogative words: *who*, *when*, *how*, *why*, *which*, *where*, and *what*. Each question is represented by a one-hot vector based on the presence of these words. For example, a question beginning with *who* is represented as [0,1,0,0,0,0,0]. The vocabulary for question types is V={"what", "who", "when", "why", "where", "which", "how"}.

**Question Category**

We use distributed semantic representations of words (i.e., word2vec) to measure the similarity between the categories of input and related questions. Here, "question category" refers to the group of questions that belong to the same thematic label. The input question categories obtained by using a question categorization module. We are given the dataset $Q$ includes question - answer pairs extracted from cQA sites, in which each question is assigned to a category label. The question categorization module aims to classify the input question $q^*$ into one of the question categories in the dataset $Q$. To this end, we implement the following steps:

1. We prepare a training dataset including questions in dataset $Q$, they are assigned with category labels (the label here is question category).

2. The questions are represented as vectors.

3. A machine learning method is used (here we choose SVM) to learn the classifier.

4. For each input question, we first represent it by feature vectors and use the classifier obtained at the third step to predict the label (i.e. question category).

Finally, the similarity score between the input question category and the related question category is calculated by the formula 1.

$$cosin\_similarity(u, v) = \frac{\sum_{i=1}^{n} u_i * v_i}{\sqrt{\sum_{i=1}^{n} (u_i)^2} * \sqrt{\sum_{i=1}^{n} (v_i)^2}} \quad (1)$$

where $u$ and $v$ are two n-dimensional vectors, $u_i$ is the $i^{th}$ element of $u$ vector.

**Word Embeddings**

Word embeddings provide vector representations of words in a continuous space, where semantic similarity between words is reflected in geometric proximity. These embeddings are learned from word co-occurrence statistics in large corpora.

In this work, we use the continuous Skip-gram model [24] of the word2vec toolkit to generate vector representations of the words. First, all the sentences in input questions, related questions and answers are tokenized and the words are then converted to vectors using the pre-trained word2vec model. In order to construct the question vector and answer vector we implement the following steps:

- Each question or answer with length $t$ is represented by a word vector ($w_1$; $w_2$; …; $w_t$), where $w_i$ is word vector representation of $i^{th}$ word. Suppose that we need to calculate the similarity between the input question q* and the answer $a_i$. Where question q* and answer $a_i$ are represented as follows: q* = ($w_1$, $w_2$, …, $w_n$) and $a_i$ = ($v_1$, $v_2$, …, $v_h$)

- For each word vector $w_i$ in q*, we will find the most similarity word vector $v_j$ in $a_i$ according to cosine measurement as in the formula 2 as below.

$$score(w_i) = \max_{1 \le j \le h} \left( cosine\_sim(w_i, v_j) \right) \quad (2)$$

where: $h$: the number of words in answer $a_i$, $w_i$: word vector representation of $i^{th}$ word in question $q^*$, $v_j$: word vector representation of $j^{th}$ word in answer $a_i$, $cosin\_similarity(w_i, v_j)$: is the cosine similarity of two vector representations of $i^{th}$ word in question $q^*$ with the $j^{th}$ word in answer $a_i$. Finally, the similarity score between input question $q^*$ and answer $a_i$ is calculated by the formula 3.

$$similarity(q^*, a_i) = \frac{\sum_{k=1}^{n} score(w_i)}{n} \quad (3)$$

Where $n$ is the number of words in question $q^*$.

*C. A Hybrid Model for Identifying Similar Questions*

In this section, we present a hybrid model that combines the BLSTM-based architecture with external knowledge features, as described in Section 3.2. The general architecture of the hybrid model is illustrated in Figure 3. The hybrid model follows the same procedure as the BLSTM-based model introduced in Section 3.1. However, in this model the input to the Multi-Layer Perceptron (MLP) is augmented with additional information derived from external knowledge. This integration provides richer contextual signals for capturing relationships between input and related questions, thereby improving similarity estimation.

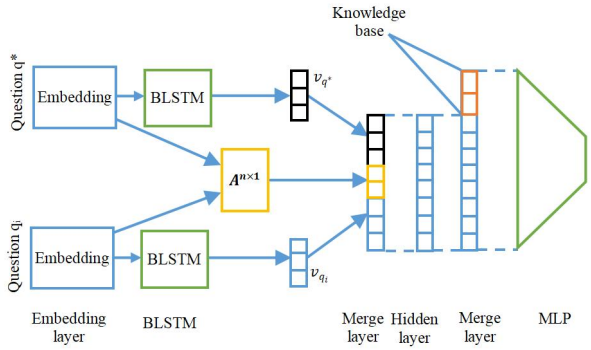


*Figure 3. The architecture of hybrid model for calculating the similarity score between input question and related question*

In this model we add vector $\boldsymbol{A^{n\times1}}$ to the merged layer. Vector $\boldsymbol{A^{n\times1}}$ is the result of calculating the similarity between the two question matrix embedding E ∈ $R^{n\times d}$. Suppose the question embedding matrix of input question q* and related question $q_i$ is $\boldsymbol{E^1} = (\boldsymbol{e_1^1}, \boldsymbol{e_2^1}, \ldots, \boldsymbol{e_n^1}) \in \boldsymbol{R^{n\times d}}$ and $\boldsymbol{E^2} = (\boldsymbol{e_1^2}, \boldsymbol{e_2^2}, \ldots, \boldsymbol{e_n^2}) \in \boldsymbol{R^{n\times d}}$, respectively. Then, vector $\boldsymbol{A^{n\times1}}$ is calculated by the formula 4:

$$A_i = \max_{1\le j\le n}\left(sim(e_i^1, e_j^2)\right) \quad (4)$$

Figure 4 illustrates the calculation of vector A from the embedding layer.

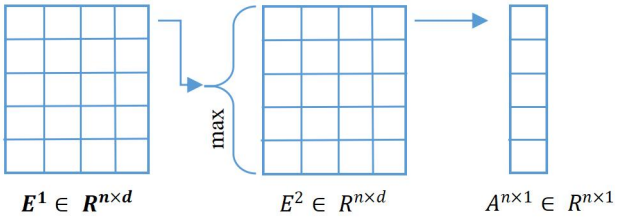


*Figure 4. Illustrates the calculation of vector A from the embedding layer*

## IV. SemEval 2016 Experiments

In this section, we will describe the experimental setup and the results on the SemEval-2016 task 3 dataset.

### *A. Dataset and Evaluation Metrics*

To evaluate our proposed model, we conducted experiments on the cQA dataset provided by SemEval-2016 Task 3, Subtask B[1]. The dataset was extracted from *Qatar Living* (http://www.qatarliving.com/forum), a web forum where users post questions on various aspects of daily life in Qatar. The dataset consists of 337 input questions and 3369 related questions. It is pre-split into 267 input questions and 2669 related questions for training, and 70 input questions with 700 related questions for testing. Each data point is a pair consisting of an input question and a related question, annotated with a similarity label: *Relevant* (1) or *Irrelevant* (0). The task is to predict the binary label and to rank related questions by their similarity with respect to the input question. Table 1 summarizes the statistics of the SemEval-2016 dataset.

*Table 1. The statistics of SemEval 2016 dataset*

| | Train | Test | Total |
|---|---|---|---|
| Input question | 267 | 70 | 337 |
| Related question-answer pairs | 2669-26690 | 700-7000 | 3369-33690 |

We used several measures to evaluate our models, they consist of: the classification measures include: *Accuracy* (*Acc*), *Precision* (*P*), *Recall* (*R*), and $F_1$ - *measure* ($F_1$); the ranking measures include: *Mean Average Precision* (*MAP*), *Average Recall* (*AvgRec*) and *Mean Reciprocal Rank* (*MRR*).

### *B. Setup*

The models in this paper are implemented with Theano[2] from scratch. We use the accuracy on the validation set to locate the best epoch and best hyper-parameter settings for testing.

The word embeddings are pre-trained using Gensim word2vec tool[3]. The training data for the word embeddings is the dataset provided by SemEval-2016. The parameters are set as follows: (1) the word vector size is 200; (2) the maximum distance between the current word and the predicted word in a sentence is set to 5; (3) ignore all words with a total frequency of less than 5. Also, we use the 300-dimensional vector trained and provided by word2vec[4].

We train our models in mini-batches (the batch size is 64), and the maximum length of input questions and related questions is 40. Any tokens out of this range will be discarded.

The hyper-parameters of BLSTM model are set as follows: memory size is set to 100, the learning rate is 0.025 and dropout rate is 0.3. We employ *SGD* (Stochastic Gradient Descent) as the optimization method and *mean squared error* as loss function for our model.

### *C. Results and Discussions*

Table 2 summarizes the performance of our models on the SemEval-2016 dataset. For the BLSTM-based model, we experimented with two types of word embeddings: word2vec vectors of size 200 and pre-trained vectors of size 300. The results show that the change in vector dimensionality did not lead to significant improvements. Specifically, the BLSTM model with 200-dimensional embeddings achieved an F1 score of 57.81 and MAP of 73.86, while the 300-dimensional variant slightly improved the F1 score to 61.64 but produced a lower MAP (71.27). These findings suggest that embedding dimensionality alone does not substantially influence performance under this dataset setting, possibly due to the

[1] http://alt.qcri.org/semeval2016/task3/index.php?id=data-and-tools

[2] http://deeplearning.net/software/theano/#

[3] www.radimrehurek.com/gensim

[4] https://code.google.com/p/word2vec/

relatively small and noisy nature of the data. In contrast, the proposed hybrid model achieved clear gains over the BLSTM baseline in both embedding settings. With 200-dimensional word2vec embeddings, the hybrid model improved the F1 score to 74.04 and MAP to 78.37**.** Similarly, with 300-dimensional embeddings, it achieved an F1 score of 72.32 and MAP of 78.38. These results demonstrate that augmenting BLSTM representations with external knowledge features provides consistent benefits across different embedding sizes.

Table 2. Summary of results on the SemEval-2016 dataset

| Model | Embedding | Classification measures | | | | Ranking measures | | |
|---|---|---|---|---|---|---|---|---|
| | | Acc | P | R | F1 | MAP | AveRec | MRR |
| BLSTM | Word2vec | 74.14 | 53.22 | 63.27 | 57.81 | 73.86 | 87.08 | 80.51 |
| BLSTM | vocabulary size | 76.00 | 57.94 | 65.85 | 61.64 | 71.27 | 88.02 | 76.75 |
| Hybrid Model | Word2vec | 83.57 | 70.39 | 78.10 | 74.04 | 78.37 | 91.97 | 86.23 |
| Hybrid Model | vocabulary size | 83.43 | 76.82 | 68.32 | 72.32 | 78.38 | 92.01 | 86.23 |

To further assess the effectiveness of our approach, we compared our hybrid model against several well-known methods reported in the literature on the same dataset (Table 3). The results show that our model achieves higher performance on both classification and ranking measures. In particular, our model obtained the highest Accuracy (83.43%), outperforming ConvKN [1] (78.71%), KeLP [2] (79.43%), and UH-PRHLT [16] (76.57%). In terms of ranking metrics, our model achieved a MAP of 78.38 and MRR of 86.23, which are superior to the majority of competing systems. These findings confirm the effectiveness of our hybrid neural model, demonstrating that integrating external features into BLSTM-based representations yields measurable improvements over both traditional feature-based approaches and purely neural models.

*Table 3. Comparison with previous studies for the same task and the same dataset*

| Models | Classification measures | | | | Ranking measures | | |
|---|---|---|---|---|---|---|---|
| | Acc | P | R | F1 | MAP | AvgRec | MRR |
| UH-PRHLT-primary [15] | 76.57 | 63.53 | 69.53 | 66.39 | 76.70 | 90.31 | 83.02 |
| ConvKN-primary [1] | 78.71 | 68.58 | 66.52 | 67.54 | 76.02 | 90.70 | 84.64 |
| Kelp-primary [2] | 79.43 | 66.79 | 75.97 | 71.08 | 75.83 | 91.02 | 82.71 |
| SLS-primary [24] | 79.43 | 76.33 | 55.36 | 64.18 | 75.55 | 90.65 | 84.64 |
| ICL00-primary [25] | 33.29 | 100 | 49.95 | 33.29 | 75.11 | 89.33 | 83.02 |
| ECNU-primary [26] | 72.71 | 100 | 18.03 | 30.55 | 73.92 | 89.07 | 81.48 |
| UniMelb-primary [27] | 74.57 | 63.96 | 54.08 | 58.60 | 70.20 | 86.21 | 78.58 |
| Our model | 83.43 | 76.82 | 68.32 | 72.32 | 78.38 | 92.01 | 86.23 |

## V. Quora Experiments

In this section, we detail our experimental setup and results using the Quora dataset.

### *A. Dataset*

The second dataset used to evaluate our approach is the Quora Question Pairs dataset[5], extracted from *https://www.quora.com/*. Quora is a large community-driven forum where users post, answer, and edit questions across a wide variety of topics. The dataset contains 404082 question pairs, consisting of input and related questions annotated with a binary similarity label (1 for semantically equivalent, 0 for non-equivalent). The dataset is pre-split into 363665 pairs for training and 40417 pairs for testing. Table 4 shows an example of some question pairs and Table 5 presents the dataset statistics, showing that the average question length is around 11 words, corresponding to approximately 60 characters.

*Table 4. An example of some question pairs of Quora dataset*

| Input question | Related question | Label |
|---|---|---|
| How do you become both a lawyer and a doctor? | How can you become a lawyer? | 0 |
| How do I get rid of the smell from a cat spraying? | How can I stop my cat from spraying? | 0 |
| How does one start a small business? | How can I start a successful small business? | 1 |
| Which are the best GMAT coaching institutes in Delhi/NCR? | What is the best coaching institute for GMAT in Delhi NCR region? | 1 |
| What are some good jobs for civil engineer? | Which are the best jobs in civil engineering? | 1 |
| What does a product developer do? | What is product development? | 0 |

*Table 5. The statistics of Quora dataset*

| | Question pairs | The average number of words | The average number of characters |
|---|---|---|---|
| Train | 363665 | 11.17 | 60.11 |
| Test | 40417 | 11.03 | 60.05 |

### *B. Setup*

For experiments on the Quora dataset, we maintained most configurations from Section 4.2, with the following modifications: First, we set the batch size as 128; Second, we set the maximum length of input questions and related questions as 20 instead of 40. Third, the training data for the word embeddings is a Quora corpus. This corpus

[5] https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs

containing about 35,65 million words, and 77,845 unique words. We also use 300-dimensional vectors that were trained and provided by word2vec using a part of the Google News dataset.

*C. Results and Discussions*

Table 6 summarizes the experimental results on the Quora dataset. The BLSTM baseline achieved accuracies of 78.92% (word2vec embeddings) and 79.40% (vocabulary-based embeddings). The F1 scores were 68.25 and 72.19, respectively. These results indicate that BLSTM is able to learn meaningful representations of questions; however, its performance remains limited when relying solely on distributed embeddings. By contrast, the proposed hybrid model achieved substantial improvements. With word2vec embeddings, the hybrid model reached an accuracy of 87.55% and an F1 score of 80.49. Using vocabulary-based embeddings, it achieved an accuracy of 87.79% and an F1 score of 80.41. These results represent a notable performance gain of approximately 8–9% in accuracy compared to the BLSTM baseline. The consistent improvements across precision, recall, and F1 suggest that incorporating additional features beyond BLSTM representations enables the hybrid model to better capture semantic relationships between question pairs. This demonstrates the robustness of our approach across different embedding sources and highlights its effectiveness on large-scale, real-world data.

*Table 6. Summary of main results used the Quora dataset*

| Model | Embedding | Acc | P | R | F1 |
|---|---|---|---|---|---|
| BLSTM | Word2vec | 78.92 | 65.01 | 71.84 | 68.25 |
| BLSTM | vocabulary size | 79.40 | 68.17 | 76.72 | 72.19 |
| Hybrid Model | Word2vec | 87.55 | 73.68 | 88.69 | 80.49 |
| Hybrid Model | vocabulary size | 87.79 | 71.87 | 91.24 | 80.41 |

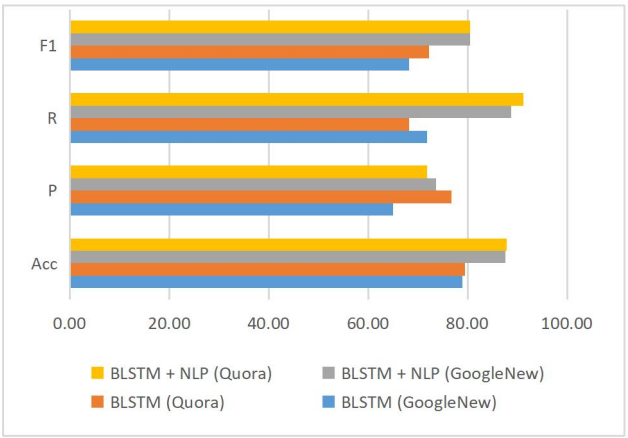


*Figure 5. Compare the performance of different models for question similarity*

## VI. Conclusion

This paper proposed a hybrid neural model for detecting and ranking similar questions in Community Question Answering (cQA) systems. The model integrates BLSTM-based representations with auxiliary external knowledge to improve similarity estimation. Experiments on SemEval-2016 Task 3 and Quora Question Pairs demonstrated consistent improvements over the BLSTM baseline, with our hybrid model achieving competitive performance compared to recent state-of-the-art methods. Overall, the results confirm the effectiveness of combining neural representations with complementary external features. Future work will focus on ablation studies and the integration of advanced pre-trained language models.

## References

[1] Alberto Barron-Cedeno, Daniele Bonadiman, Giovanni Da San Martino. ConvKN at SemEval-2016 Task 3: Answer and Question Selection for Question Answering on Arabic and English Fora. In Proceedings of SemEval-2016, pp. 896–903 (2016)

[2] Simone Filice, Danilo Croce, et al. KeLP at SemEval-2016 Task 3: Learning Semantic Relations between Questions and Answers. In Proceedings of SemEval-2016, pp. 1116–1123 (2016)

[3] Kai Wang, Zhaoyan Ming, Tat-Seng Chua. A syntactic tree matching approach to finding similar questions in community-based qa services. In SIGIR, pp. 187-194 (2009)

[4] Wen-tau Yih, Xiaodong He, Christopher Meek. Semantic Parsing for Single-Relation Question Answering. In Proceedings of ACL, pp. 643-648 (2014)

[5] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, Gregoire Mesnil. Learning Semantic Representations Using Convolutional Neural Networks for Web Search. In Proceedings of the 23rd International Conference on World Wide Web, pp. 373-374 (2014)

[6] Nal Kalchbrenner, Edward Grefenstette, Phil Blunsom. A convolutional neural network for modelling sentences. In Proceedings of ACL, pp 655–665, (2014)

[7] Ye Zhang, Byron C. Wallace. A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification. arXiv:1510.03820v4 [cs.CL] 6 Apr (2016)

[8] Yunbo Cao, Huizhong Duan, Chin-Yew Lin, Yong Yu, Hsiao-Wuen Hon. Recommending Questions Using the Mdl-based Tree Cut Model. In Proceedings of WWW, pp. 81-90 (2008)

[9] Huizhong Duan, Yunbo Cao, Chin-Yew Lin, Yong Yu. Searching Questions by Identifying Question Topic and Question Focus. In Proceedings of ACL-08: HLT, pp.156-164 (2008)

[10] Zongcheng Ji, Fei Xu, Bin Wang, Ben He. Question-answer topic model for question retrieval in community question answering. ACM, pp. 2471-2474 (2012)

[11] Kai Zhang, Wei Wu, Haocheng Wu, Zhoujun Li, Ming Zhou. Question retrieval with high quality answers in community question answering. In Proceedings of ACM, pp. 371-380 (2014)

[12] Guangyou Zhou, Li Cai, Jun Zhao, Kang Liu. Phrase-based translation model for question retrieval in community question answer archives. In Proceedings of ACL, pp. 653-662 (2011)

[13] Jiwoon Jeon, W. Bruce Croft, Joon Ho Lee. Finding Similar Questions in Large Question and Answer Archives. In Proceedings of ACM, pp. 84-90 (2005)

[14] Simone Filice, Giuseppe Castellucci, Danilo Croce, Roberto Basili. KeLP: a Kernel-based Learning Platform for Natural Language Processing. In Proceedings of ACL-IJCNLP, pp. 19-24 (2015)

[15] Marc Franco-Salvador, Sudipta Kar, Thamar Solorio, and Paolo Rosso. UH-PRHLT at SemEval-2016 Task 3: Combining Lexical and Semantic-based Features for Community Question Answering. In Proceedings of SemEval-2016, pp. 814–821 (2016)

[16] Yann LeCun, Yoshua Bengio, Geoffrey Hinton. Deep learning. Nature, 521(7553), pp. 436–444 (2015)

[17] Yoav Goldberg. A primer on neural network models for natural language processing. arXiv preprint arXiv:1510.00726, (2015)

[18] Alex Graves. Supervised Sequence Labelling with Recurrent Neural Networks. SCI, vol. 385. Springer, Heidelberg (2012)

[19] Ming Tan, Bing Xiang, Bowen Zhou. Lstm-based deep learning models for non-factoid answer selection. arXiv:1511.04108 [cs.CL]. (2015)

[20] Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. Applying deep learning to answer selection: A study and an open task. CoRR, abs/1508.01585. (2015)

[21] Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. Deep learning for answer sentence selection. CoRR, abs/1412.1632. (2014)

[22] Cicero dos Santos, Luciano Barbosa, Dasha Bogdanova, Bianca Zadrozny. Learning hybrid representations to retrieve semantically equivalent questions. In Proceedings of ACL, pp. 694–699 (2015)

[23] Dzmitry Bahdanau, KyungHyun Cho, and Yoshua. Bengio. Neural machine translation by jointly learning to align and translate. In Proceedings of International conference of learning representations. arXiv:1409.0473 [cs.CL] (2015)

[24] Mitra Mohtarami, Yonatan Belinkov, Wei-Ning Hsu, Yu Zhang et al. SLS at SemEval-2016 Task 3: Neural-based Approaches for Ranking in Community Question Answering. In Proceedings of SemEval-2016, pp. 828–835 (2016)

[25] Minghua Zhang, Yunfang Wu. ICL00 at SemEval-2016 Task 3: Translation-Based Method for CQA System. In Proceedings of SemEval-2016, pp. 857–860 (2016)

[26] Guoshun Wu, Man Lan. ECNU at SemEval-2016 Task 3: Exploring Traditional Method and Deep Learning Method for Question Retrieval and Answer Ranking in Community Question Answering. In Proceedings of SemEval-2016, pp. 872–878 (2016)

[27] Doris Hoogeveen, Yitong Li, et al. UniMelb at SemEval-2016 Task 3: Identifying Similar Questions by Combining a CNN with String Similarity Measures. In Proceedings of SemEval-2016, pp. 851–856 (2016)