# Convolutional Neural Tensor Network Architecture for Community-based Question Answering

**Xipeng Qiu and Xuanjing Huang**
Shanghai Key Laboratory of Data Science, Fudan University
School of Computer Science, Fudan University
825 Zhangheng Road, Shanghai, China
xpqiu@fudan.edu.cn, xjhuang@fudan.edu.cn

## Abstract

Retrieving similar questions is very important in community-based question answering. A major challenge is the lexical gap in sentence matching. In this paper, we propose a convolutional neural tensor network architecture to encode the sentences in semantic space and model their interactions with a tensor layer. Our model integrates sentence modeling and semantic matching into a single model, which can not only capture the useful information with convolutional and pooling layers, but also learn the matching metrics between the question and its answer. Besides, our model is a general architecture, with no need for the other knowledge such as lexical or syntactic analysis. The experimental results shows that our method outperforms the other methods on two matching tasks.

## 1 Introduction

Community-based (or collaborative) question answering (CQA) such as Yahoo! Answers[1] and Baidu Zhidao[2] has become a popular online service in recent years. Unlike traditional question answering (QA), information seekers can post their questions on a CQA website which are later answered by the other users. However, with the increase of the CQA archive, it accumulates massive duplicated questions on the CQA websites. One of the primary reasons is that information seekers cannot retrieve answers they need and thus post another new question consequently. Therefore, it becomes more and more important to find semantically similar questions.

The major challenge for CQA retrieval is the problem of lexical gap (or lexical chasm) among the questions [Jeon *et al.*, 2005; Xue *et al.*, 2008]. Since question-answer (QA) pairs are relatively short, the word mismatching problem is especially important, as shown in Table 1.

| |
|---|
| **Query**:<br>Q: Why is my laptop screen blinking? |
| **Expected**:<br>Q1: How to troubleshoot a flashing screen on an LCD monitor? |
| **Not Expected**:<br>Q2: How to make text blink on screen with PowerPoint? |

Table 1: An example on question retrieval

The state-of-the-art studies [Blooma and Kurian, 2011] mainly focus on finding textual clues to improve the similarity function, such as translation-based [Xue *et al.*, 2008; Zhou *et al.*, 2011] or syntactic-based approaches [Wang *et al.*, 2009; Carmel *et al.*, 2014]. However, the improvements of these approaches are limited. The reason is that it is difficult to design a good similarity function for the discrete representations of words.

Recently, various methods are proposed to learn distributed representations of words (word embeddings) in a low-dimensional vector space. Distributed representations help learning algorithms to achieve better performance by grouping similar words, and have been extensively applied on many natural language processing (NLP) tasks [Turian *et al.*, 2010; Mikolov *et al.*, 2010; Collobert *et al.*, 2011].

In this paper, we propose a novel unified model for CQA, **convolutional neural tensor network** (CNTN), which integrates the sentence modeling and semantic matching into a single model. Specifically, we first transform all the word tokens into vectors by a lookup layer, then encode the questions and answers to fixed-length vectors with convolutional and pooling layers, and finally model their interactions with a tensor layer. Thus, our model can group similar questions and answers in a semantic vector space and avoid the problem of lexical gap. The topmost tensor layer can be regarded as a kind of metric learning methods [Xing *et al.*, 2002] to measure the relevance of two texts, and learn a better metric than the traditional similarity metrics, such as inner-product or Euclidean distance.

The contributions of this paper can be summarized as fol-

[1] http://answers.yahoo.com/
[2] http://zhidao.baidu.com/

lows.

1. Our proposed CNTN architecture integrates the sentence modeling and semantic matching into a unified model, which can not only capture the useful semantic and structure information in convolutional and pooling layers, but also learn the matching metrics between texts in the topmost tensor layer.
2. CNTN is a general architecture and need not the complicated NLP pre-processing (such as syntactic analysis) or prior knowledge (such as WordNet).
3. We perform extensive empirical studies on two matching tasks, and demonstrate that CNTN is more effective than the other models.

## 2 Related Works

### 2.1 Questions Retrieval

In CQA, various techniques have been studied to solve lexical gap problems for question retrieval. The early works can be traced back to finding similar questions in Frequently Asked Questions (FAQ) archives, such as the FAQ finder [Burke *et al.*, 1997], which usually used statistical and semantic similarity measures to rank FAQs.

Jeon *et al.*[2005] showed that the translation model outperforms the others. In subsequent works, some translation-based methods [Xue *et al.*, 2008; Zhou *et al.*, 2011] were proposed to more sophisticatedly combine the translation model and the language model for question retrieval. Although these methods has yielded the state-of-the-art performance for question retrieval, they model the word translation probabilities without taking into account the structure of whole sentences.

Another kind of methods [Wang *et al.*, 2009; Carmel *et al.*, 2014] utilized the question structures to improve the similarity in question matching. However, these methods depend on a external parser to get the grammar tree of a sentence.

### 2.2 Neural Sentence Model

With the recent development of deep learning, most methods [Bengio *et al.*, 2003; Mikolov *et al.*, 2010; Collobert *et al.*, 2011] are primarily focus on learning the distributed word representations (also called **word embeddings**).

Beyond words, there are some other methods to model the sentence, called **neural sentence models**. The primary role of the neural sentence model is to represent the variable-length sentence as a fixed-length vector. These models generally consist of a projection layer that maps words, subword units or n-grams to high dimensional embeddings (often trained beforehand with unsupervised methods); the latter are then combined with the different architectures of neural networks, such as Neural Bag-of-Words (NBOW), recurrent neural network (RNN), recursive neural network(RecNN), convolutional neural network (CNN) and so on.

A simple and intuitive method is the Neural Bag-of-Words (NBOW) model. However, a main drawback of NBOW is that word order is lost. Although NBOW is effective for general document classification, it is not suitable for short sentences. A sentence model based on a recurrent neural network is sensitive to word order, but it has a bias towards the latest words that it takes as input [Mikolov *et al.*, 2010]. This gives the RNN excellent performance at language modelling, but it is suboptimal for modeling the whole sentence. [Le and Mikolov, 2014] proposed Paragraph Vector (PV) to learn continuous distributed vector representations for pieces of texts, which can be regarded as a long term memory of sentence as opposed to short memory in RNN.

Recursive neural network (RecNN) adopts a more general structure to encode sentence [Pollack, 1990; Socher *et al.*, 2013b]. At every node in the tree the contexts at the left and right children of the node are combined by a classical layer. The weights of the layer are shared across all nodes in the tree. The layer computed at the top node gives a representation for the sentence. However, RecNN depends on external constituency parse trees provided by an external parse tree.

Convolutional neural network (CNN) is also used to model sentences [Kalchbrenner *et al.*, 2014; Hu *et al.*, 2014]. As illustrated in Figure 1, it takes as input the embedding of words in the sentence aligned sequentially, and summarizes the meaning of a sentence through layers of convolution and pooling, until reaching a fixed length vectorial representation in the final layer. CNN has some advantages: (1) it can maintain the word order information which is crucial to the short sentences; (2) Nonlinear activation in the convolutional neural networks can learn more abstract characteristics.

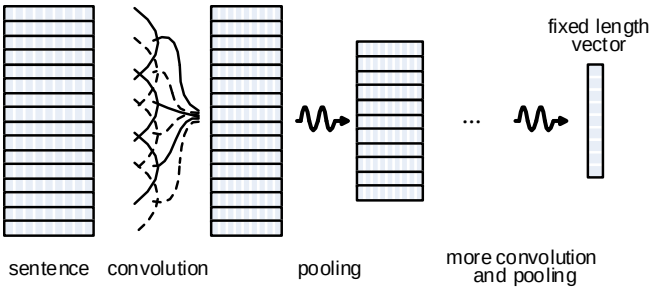



Figure 1: Sentence modelling with convolutional neural network.

## 3 Modeling Question and Answers with Convolutional Neural Network

In this paper, we use CNN to encode the sentence. The original CNN can learn sequence embeddings in a supervised way. In our model, the parameters in CNN are learnt jointly with our final objective function instead of separate training.

Given an input sentence $s$, we take the embeddings $\mathbf{w}_i \in \mathbb{R}^{n_w}$ of each word $w$ in $s$ to obtain the first layer of the CNN.

**Convolution** The embeddings for all words in the sentence $s$ construct the input matrix $\mathbf{s} \in \mathcal{R}^{n_w \times l_s}$, where $l_s$ denotes the length of $s$. A convolutional layer is obtained by convolving a matrix of weights (filter) $\mathbf{m} \in \mathcal{R}^{n \times m}$ with the matrix of activations at the layer below, where $m$ is the filter width.

For example, the first layer is obtained by applying a convolutional filter to the sentence matrix $\mathbf{s}$ in the input layer. Dimension $n_w$ and filter width $m$ are hyper-parameters of the network.

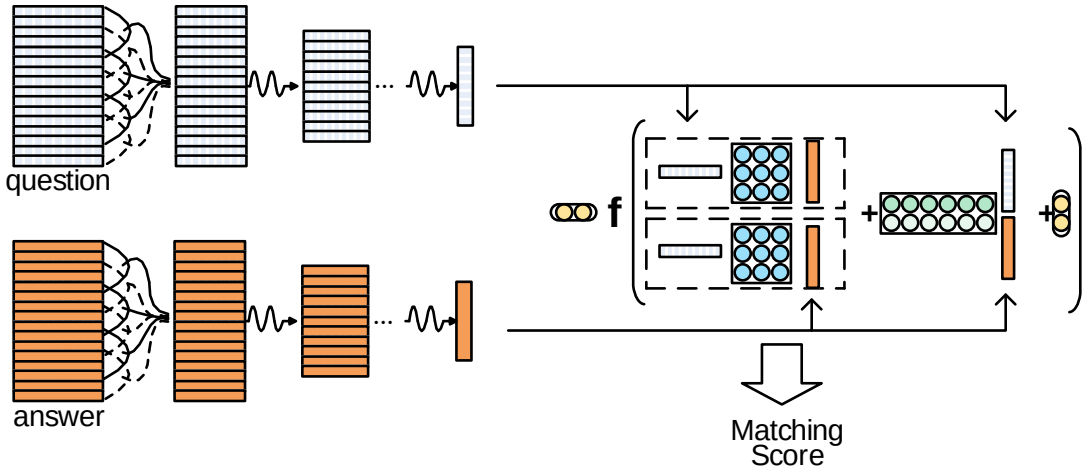



Figure 2: Architecture of Neural Tensor Network.

**$k$-Max Pooling** Given a value $k$ and a row vector $\mathbf{p} \in \mathcal{R}^p$, $k$-max pooling selects the subsequence $\mathbf{p}^k_{max}$ of the $k$ highest values of $\mathbf{p}$. The order of the values in $p^k_{max}$ corresponds to their original order in $\mathbf{p}$. The $k$-max pooling operation makes it possible to pool the $k$ most active features in $\mathbf{p}$. The pooling layers preserve the order of the features, but are insensitive to the specific positions of input sequences.

To fix the total number of convolutional layers, we use a dynamic $k$-max pooling [Kalchbrenner *et al.*, 2014] to set $k$ be a function of the length of the sentence and the depth of the network $d$.

The $k_{top} = n_s$ is fixed at the $k$-max pooling layer after the topmost convolutional layer, where $n_s$ is the length of sentence embeddings. This guarantees that the input of the fully connected layers is independent of the length of the input sentence. However, at the intermediate convolutional layers, the pooling parameter $k$ is not fixed, and it is dynamically selected in order to allow for a smooth extraction of higher order and longer-range features.

**Non-linear Feature Function** After (dynamic) k-max pooling is applied to the result of a convolution, a bias and a non-linear function $tanh$ are applied component-wise to the pooled matrix.

**Final Layer** Different with the original CNN [Kalchbrenner *et al.*, 2014], the final output of our CNN is a fixed-length vector, which represents the embeddings $\mathbf{v}_s \in \mathcal{R}^{n_s}$ of the input sentence $s$. The parameters in CNN is learnt jointly with our final objective function.

## 4 Matching Question and Answer with Tensor Layer

To model the interactions between question and answer, we need utilize some metrics to measure their relevance. Given their vector representations, the traditional ways are to calculate their Euclidean or cosine distance. However, these two ways cannot sufficiently take into account the complicated interactions.

In this paper, we model the matching degree of two sentences with a non-linear tensor layer, which has been successfully applied to explicitly model multiple interactions of relational data [Socher *et al.*, 2013a]. A tensor is a geometric object that describes relations between vectors, scalars, and other tensors. It can be represented as a multi-dimensional array of numerical values.

Given a question $q$ and its corresponding answer $a$, we first use two CNNs to model them into the fixed vectors $\mathbf{q}$ and $\mathbf{a}$ respectively. Following the Neural Tensor Network (NTN) [Socher *et al.*, 2013a], we place a tensor layer on top of the two CNNs (introduced in Section 3) to model the relations of the question and its answer. Figure 2 shows a visualization of our general architecture.

The tensor layer calculates the matching degree of a question-answer pair by the following score function:

$$s(q,a) = \mathbf{u}^T \mathbf{f}\left(\mathbf{v}_q^T \mathbf{M}^{[1:r]} \mathbf{v}_a + \mathbf{V}\begin{bmatrix}\mathbf{v}_q \\ \mathbf{v}_a\end{bmatrix} + \mathbf{b}\right), \quad (1)$$

where $\mathbf{f}$ is a standard nonlinearity applied element-wise, $\mathbf{M}^{[1:r] \in \mathcal{R}^{n_s \times n_s \times r}}$ is a tensor and the bilinear tensor product $\mathbf{v}_q^T \mathbf{M}^{[1:r]} \mathbf{v}_a$ results in a vector $h \in \mathcal{R}^r$, where each entry is computed by one slice $i = 1, \cdots, r$ of the tensor: $h_i = \mathbf{v}_q^T \mathbf{M}^i \mathbf{v}_a$; the other parameters are the standard form of a neural network: $\mathbf{V} \in \mathcal{R}^{r \times 2n_s}$, $\mathbf{b} \in \mathcal{R}^r$ and $\mathbf{u} \in \mathcal{R}^r$.

We call the whole architecture of our model **convolutional neural tensor network** (CNTN). The main advantage of CNTN is that it can model the representations and interactions jointly. The representations of words and sentences are modeled with convolutional layers, and the interactions between two sentences are modeled with the tensor layer. The final output is the matching score of two texts.

## 5 Training

We use the contrastive max-margin criterion [Bordes *et al.*, 2013; Socher *et al.*, 2013a] to train our model. Intuitively, the

max-margin criterion provides an alternative to probabilistic, likelihood-based estimation methods by concentrating directly on the robustness of the decision boundary of a model [Taskar *et al.*, 2005]. The main idea is that each question-answer pair in the training set $\mathcal{C}$ should receive a higher score than a question-answer pair in which the answer is replaced with a random answer of the other questions.

The parameters of our model are $\Theta = \{\mathbf{L}, \mathbf{W}^q_{CNN}, \mathbf{W}^a_{CNN}, \mathbf{u}, \mathbf{M}^{[1:r]}, \mathbf{V}, \mathbf{b}\}$. Among them, $\mathbf{L}$ is words embeddings; $\mathbf{W}^q_{CNN}$ and $\mathbf{W}^a_{CNN}$ are the parameters of the CNNs for question and answer respectively; the rest parameters belong to the tensor layer.

The final objective function is

$$\mathcal{L} = \sum_{(q,a)\in\mathcal{C}} \sum_{(q,a')\in\mathcal{C}'} [\gamma - s(q,a) + s(q,a')]_+ + \lambda||\Theta||_2^2 \quad (2)$$

where $\gamma > 0$ is a margin hyper-parameter, and $(q, a')$ is a corrupted question-answer pair. $\mathcal{C}$ is the training collection of question-answer pairs and $\mathcal{C}'$ denotes the collection of all corrupted question-answer pairs. We use Eq. 1 as the score function $s(\cdot,\cdot)$ to balance the efficiency and performance of the algorithm.

To minimize the objective, we use stochastic gradient descent(SGD) with the diagonal variant of AdaGrad [Duchi *et al.*, 2011]. The parameter update for the $i$-th parameter $\theta_{t,i}$ at time step $t$ is as follows:

$$\theta_{t,i} = \theta_{t-1,i} - \frac{\rho}{\sqrt{\sum_{\tau=1}^{t} g_{\tau,i}^2}} g_{t,i}, \quad (3)$$

where $\rho$ is the initial learning rate and $g_\tau \in \mathbb{R}^{|\theta_i|}$ is the sub-gradient at time step $\tau$ for parameter $\theta_i$.

# 6 Experiments

To empirically demonstrate the effectiveness of our approach, we use two datasets from different languages (English and Chinese) in our experiments.

**English** For English, we collect the question-answer pairs from Yahoo! Answers with the getByCategory function provided in Yahoo! Answers API. More specifically, we utilize the resolved questions under the top-level "Computers & Internet" category. Finally, the English dataset contains $312,000$ question-answer pairs.

**Chinese** For Chinese, we use a crawler to collect the question-answer pairs from Baidu Zhidao website. We also just use the questions under the "Computers & Internet" category. Chinese sentences are segmented into words in advance. Finally, the Chinese dataset contains $423,000$ question-answer pairs.

For the above two datasets, we only selected those resolved questions.

## 6.1 Competitors

We compare our model with the following methods:

- **Okapi**: The popular Okapi BM25 model [Robertson *et al.*, 1994] used in information retrieval.
- **TransLM**: We use translation-based language model of [Jeon *et al.*, 2005] to calculate the similarity between two short-texts. TransLM can be regarded as state-of-the-art method for question retrieval.
- **NBOW+MLP**: We first represent each short-text as the sum of the embeddings of the words it contains. The matching score of two short-texts are calculated with a multi-layer perceptron (MLP) [Bengio, 2009] with the embeddings of the two short-texts as input;
- **CNN+MLP**: We first represent each short-text with CNN, then a MLP is used to score the matching degree of the two short-texts. This method is similar with [Hu *et al.*, 2014].

The first two methods are traditional methods and use discrete word representation. The latter two methods use distributed vector to model sentences, then compare the representation for the two sentences with MLP.

For initialization of parameters, we use word2vec [Mikolov *et al.*, 2013] to train word embeddings on Wikipedia corpus for English and Chinese respectively. For the other parameters, we use random initialization within $(0.01, 0.01)$. The parameters which achieve the best performance on the development set will be chosen for the final evaluation. All the other models are trained on the same training set as our models.

## 6.2 Experiment I: Matching Question and Answers

We select 10,000 original positive pairs as development set and another 10,000 original positive pairs as test set. The rest QA pairs are used for training. For each positive QA pair in training set, we construct ten corrupted QA pairs as negative instances by replacing the answer with negative sampling.

We first evaluate our model with five different settings of the tensor layer and compare their matching abilities to pick the correct answer to the corresponding question from the five random negative answers.

- **CNTN-I**: We set the parameters $r = 1, \mathbf{M}^{[1]} = I, \mathbf{V} = 0, \mathbf{b} = 0, \mathbf{f} = \text{identity}$. The model is the most simplified model, which is just a cosine similarity.
- **CNTN-II**: We set the parameters $r = 1, \mathbf{V} = 0, \mathbf{b} = 0, \mathbf{f} = \text{identity}$. The model can be regarded as the bilinear model.
- **CNTN-III**: We set the parameters $r = 1, \mathbf{M}^{[1]} = 0$. The model can be regarded as single layer MLP.
- **CNTN-IV**: We set the parameter $r = 1$. This model is the fully CNTN model.
- **CNTN-IV**: We set the parameter $r = 5$. This model is the fully CNTN model.

The other hyperparameters of our model are set as in Table 2, which are chosen on development datasets in consideration of both accuracy and efficiency.

Table 3 shows the accuracy(P@1) of the above five models on English and Chinese development sets. The fully CNTN model with $r = 5$ (CNTN-V) achieves best performances.

| | |
|---|---|
| Word embedding size | $n_w = 25$ |
| Initial learning rate | $\rho = 0.1$ |
| Regularization | $\lambda = 10^{-4}$ |
| CNN depth | $d = 3$ |
| Filter width | $m = 3$ |
| Sentence embedding size | $n_s = 50$ |

Table 2: Some major hyperparameters of CNTN model

| Methods | P@1 | |
|---|---|---|
| | English | Chinese |
| CNTN-I | 68.2 | 60.1 |
| CNTN-II | 70.5 | 63.7 |
| CNTN-III | 67.5 | 61.0 |
| CNTN-IV | 71.6 | 64.5 |
| CNTN-V | **75.8** | **68.5** |

Table 3: Comparisons of our model with different settings on development sets.

We also find that the bilinear model (CNTN-II) is better than the single layer MLP (CNTN-III). This verifies the hypothesis that the tensor is effective to incorporate the strong interaction of two sentences although the bilinear form can only model simpler interactions. In comparison to bilinear models, the fully CNTN has much more expressive power to model the strong interactions of two sentences.

We use the fully CNTN in the final evaluations on the test sets, and the comparisons of the other methods are shown in Table 4. CNTN achieves best performance.

### 6.3 Experiment II: Question Retrieval

The motivation of our architecture is to learn semantic representation and reduce the "lexical gap". To do so, given a queried question $t$, we first represent it in distributed representation $\mathbf{v}_t$ by DCNN. For each question-answer pair $(q, a)$ in CQA collections $\mathcal{C}$, we use a mixture score to compute its similarity with queried question $t$.

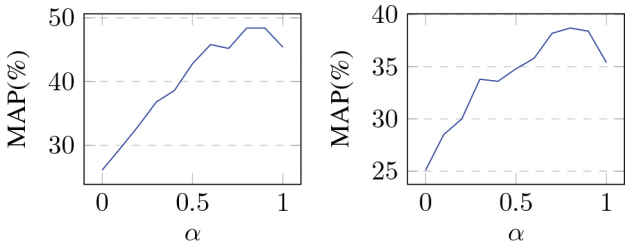
$$(\hat{q}, \hat{a}) = \arg\max_{(q,a)\in\mathcal{C}} \alpha \mathbf{v}_t^T \mathbf{v}_q + (1-\alpha) s(t, a), \qquad (4)$$

where $\mathbf{v}_q$ and $\mathbf{v}_a$ are the distributed representations of question-answer pair $(q, a)$; $s(t, a)$ is the score function of CNTN defined in Eq. 1; $\alpha \in [0, 1]$ is a hyperparameter and optimized by searching with the step-size $0.05$.

We randomly select $5,000$ QA pairs as the development set, another $5,000$ pairs as the test set. To obtain the ground-truth of question retrieval, we employ the Vector Space Model (VSM) to retrieve the top 20 results for each question in development and test set. The top 20 results do not include the queried question itself. Given a returned result by VSM, two annotators are asked to label it with "relevant" or "irrelevant". If an annotator considers the returned result semantically equivalent to the queried question, he labels it as "relevant"; otherwise, it is labeled as "irrelevant". If a conflict happens, the third annotator will make the final judgement.

| Methods | P@1 | |
|---|---|---|
| | English | Chinese |
| Okapi | 35.6 | 30.8 |
| TransLM | 48.5 | 45.3 |
| NBOW+MLP | 66.8 | 65.4 |
| CNN+MLP | 68.5 | 66.1 |
| CNTN | **70.7** | **68.1** |

Table 4: Comparisons of different models on test sets.




(a) English (b) Chinese

Figure 3: Balancing the contributions of question and answer on development sets.

| Methods | English | | Chinese | |
|---|---|---|---|---|
| | MAP | P@10 | MAP | P@10 |
| Okapi | 32.5 | 22.9 | 29.3 | 20.3 |
| TransLM | 38.6 | 25.2 | 30.7 | 22.9 |
| NBOW | 39.3 | 26.8 | 31.5 | 23.3 |
| CNN | 41.8 | 27.4 | 32.3 | 24.1 |
| CNTN | **43.9** | **28.1** | **33.5** | **24.9** |

Table 5: Question retrieval performances on test sets.

Since the fully CNTN model with $r = 5$ (CNTN-V) achieves best performances, we just use the same parameters as CNTV-V. The only difference is that we need to decide $\alpha$ in Eq. (4) on the development set.

We use the **mean average precision** (MAP) and $P@10$ to evaluate the effectiveness of each method, which are widely used in question retrieval [Jeon *et al.*, 2005; Xue *et al.*, 2008; Zhou *et al.*, 2011]. MAP rewards the methods that return relevant questions early and also rewards correct ranking of the results. $P@10$ reports the fraction of the top ten questions retrieved that are relevant.

In Eq. (4), we use the parameter $\alpha$ to balance the relative contributions of question and answer. Figure 3 illustrates how the value of $\alpha$ affects the performance of question retrieval in terms of MAP. The result was obtained with both the English and Chinese development sets. We see that our model performs best when $\alpha$ is around $0.7$, which indicates a good balance tends to benefit from question more than answer.

Finally, we compare our model with the other methods with $\alpha = 0.7$. The experiment results on test sets are illustrated in Table 5, which show that our model outperforms the others on both English and Chinese datasets.

For the above two experiments, we conducted a significance test (t-test) on the improvements of our model over the others. The result indicates that the improvements are statisti-

cally significant (p-value ¡ 0.05) in terms of all the evaluation measures.

## 7 Discussions

We can summarize some findings from these experiments:

1. The simple sum of word embeddings (NBOW) yields reasonably better results than the traditional discrete represents. The reason is that word embeddings get benefits from dense representation and reduce the negative impact of lexical gap.
2. CNN is better than NBOW, which indicates CNN can capture more informative features, such as the salient words or the sequential structures of sentences. The effectiveness of these similar features are also verified in traditional retrieval methods [Carmel *et al.*, 2014].
3. Our model (CTNT) outperforms others significantly since our model can model the complicated interactions between question and answer than others. The simple inner-product (CTNT-I) and bilinear model (CNTN-II) also cannot sufficiently model the interaction between the sentences. The importance of the interaction of two sequences is also reported in [Hu *et al.*, 2014]. Our model can be regarded as a kind of metric learning methods [Xing *et al.*, 2002] and learn a good metric to model the relevance of two sentences.
4. The performance of question retrieval can be improved by incorporating the answer part, which also provides an evidence for the effectiveness of our model in capturing the relevance information between question and answer. This is compatible with the results in [Zhou *et al.*, 2011]. They also found that it would be helpful to bridge the lexical gap by incorporating the answer information.
5. The performances on Chinese dataset are slightly worse than English. The reason behind this could be that there are some errors in Chinese word segmentation.

Compared with the other neural network based methods for matching texts pairs [Wang *et al.*, 2010; Lu and Li, 2013; Hu *et al.*, 2014], our model also aims to seek a relevance metric in a semantic vector space. However, our method can deal with more complicated interactions with the tensor layer than the other methods. Meanwhile, our convolutional layers convert the texts into fix-length vectors and keep the important sequence information which are lost in bag-of-words representations [Wang *et al.*, 2010; Lu and Li, 2013]. Different to the binary max pooling CNN in [Hu *et al.*, 2014], we use a dynamic $k$-max pooling in [Kalchbrenner *et al.*, 2014]. The binary max pooling need deeper networks, therefore it is hard to train for long sentences. The binary max pooling works well for very short texts but is not suitable for matching QA pairs since that the texts in QA pairs are relatively longer, especially in answer parts.

## 8 Conclusion

We propose a convolutional neural tensor network (CNTN) architecture to model the questions and answers in CQA. CNTN can model the complicated interactions between question and answer. It is effective in resolving the problem of lexical chasm for question retrieval.

For future research, we will extend our model to capture more detailed interactions according to the category structure of questions. We believe that the interactions should vary with the different categories or topics of questions. Moreover, we also wish to investigate the ability of our model for matching the questions and experts in answerer recommendation.

## Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments. This work was partially funded by National Natural Science Foundation of China (61472088, 61473092), The National High Technology Research and Development Program of China (2015AA015408), Shanghai Science and Technology Development Funds (13dz2260200, 13511504300, 14ZR1403200), Shanghai Leading Academic Discipline Project (B114).

## References

[Bengio *et al.*, 2003] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.

[Bengio, 2009] Yoshua Bengio. Learning deep architectures for AI. *Foundations and trends in Machine Learning*, 2(1):1–127, 2009.

[Blooma and Kurian, 2011] M.J. Blooma and J.C. Kurian. Research issues in community based question answering. In *Proceedings of PACIS*, 2011.

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, 2013.

[Burke *et al.*, 1997] R. Burke, K. Hammond, V. Kulyukin, S. Lytinen, N. Tomuro, and S. Schoenberg. Question answering from frequently asked question files: Experiences with the faq finder system. *AI Magazine*, 18(2):57–66, 1997.

[Carmel *et al.*, 2014] David Carmel, Avihai Mejer, Yuval Pinter, and Idan Szpektor. Improving term weighting for community question answering search using syntactic analysis. In *Proceedings of CIKM*, 2014.

[Collobert *et al.*, 2011] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

[Duchi *et al.*, 2011] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011.

[Hu *et al.*, 2014] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. In *Advances in Neural Information Processing Systems*, 2014.

[Jeon *et al.*, 2005] J. Jeon, W.B. Croft, and J.H. Lee. Finding similar questions in large question and answer archives. *Proceedings of the ACM international conference on Information and knowledge management*, 2005.

[Kalchbrenner *et al.*, 2014] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. In *Proceedings of ACL*, 2014.

[Le and Mikolov, 2014] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of ICML*, 2014.

[Lu and Li, 2013] Zhengdong Lu and Hang Li. A deep architecture for matching short texts. In *Advances in Neural Information Processing Systems*, 2013.

[Mikolov *et al.*, 2010] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *INTERSPEECH*, 2010.

[Mikolov *et al.*, 2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*. 2013.

[Pollack, 1990] Jordan B Pollack. Recursive distributed representations. *Artificial Intelligence*, 46(1):77–105, 1990.

[Robertson *et al.*, 1994] S.E. Robertson, S. Walker, S. Jones, M.M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *TREC*, pages 109–126, 1994.

[Socher *et al.*, 2013a] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, 2013.

[Socher *et al.*, 2013b] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, 2013.

[Taskar *et al.*, 2005] Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. Learning structured prediction models: A large margin approach. In *Proceedings of the international conference on Machine learning*, 2005.

[Turian *et al.*, 2010] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *ACL*, 2010.

[Wang *et al.*, 2009] K. Wang, Z. Ming, and T.S. Chua. A syntactic tree matching approach to finding similar questions in community-based QA services. In *Proceedings of SIGIR*, 2009.

[Wang *et al.*, 2010] Baoxun Wang, Xiaolong Wang, Chengjie Sun, Bingquan Liu, and Lin Sun. Modeling semantic relevance for question-answer pairs in web social communities. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2010.

[Xing *et al.*, 2002] Eric P Xing, Michael I Jordan, Stuart Russell, and Andrew Y Ng. Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*, 2002.

[Xue *et al.*, 2008] X. Xue, J. Jeon, and W.B. Croft. Retrieval models for question and answer archives. In *Proceedings of SIGIR*, 2008.

[Zhou *et al.*, 2011] G. Zhou, L. Cai, J. Zhao, and K. Liu. Phrase-based translation model for question retrieval in community question answer archives. In *Proceedings of ACL*, 2011.