Question/Answer Matching for CQA System via Combining Lexical and Sequential Information

Yikang Shen$^\sharp$, Wenge Rong†‡, Zhiwei Sun†, Yuanxin Ouyang†‡, Zhang Xiong†‡
$^\sharp$Sino-French Engineer School, Beihang University, Beijing 100191, China
†School of Computer Science and Engineering, Beihang University, Beijing 100191, China
‡Research Institute of Beihang University in Shenzhen, Shenzhen 518057, China
{yikang.shen, w.rong, sunzhiwei, oyyx, xiongz}@buaa.edu.cn

Abstract

Community-based Question Answering (CQA) has become popular in knowledge sharing sites since it allows users to get answers to complex, detailed, and personal questions directly from other users. Large archives of historical questions and associated answers have been accumulated. Retrieving relevant historical answers that best match a question is an essential component of a CQA service. Most state of the art approaches are based on bag-of-words models, which have been proven successful in a range of text matching tasks, but are insufficient for capturing the important word sequence information in short text matching. In this paper, a new architecture is proposed to more effectively model the complicated matching relations between questions and answers. It utilises a similarity matrix which contains both lexical and sequential information. Afterwards the information is put into a deep architecture to find potentially suitable answers. The experimental study shows its potential in improving matching accuracy of question and answer.

Introduction

Community-based Question Answering (CQA) systems are Internet services which enable users to ask questions and receive answers. By using CQA systems, users can ask questions via sentences rather than issuing queries in the form of keywords to a Web search engine. CQA is proven success for knowledge sharing since it is easier for users to express their real information needs in natural language (Bilotti et al. 2010). Furthermore, using CQA is also easier to get answers to a personal nature, extremely specific questions, and even open-ended questions as in these cases it is difficult for a search engine to directly provide such complex and heterogeneous information (Chua and Banerjee 2013). CQA sites have become a kind of popular forum for people to seek information and share knowledge. Examples of such CQA sites include Yahoo! Answers and Baidu Zhidao.

Though CQA has shown its promising applicability, there still exists several challenges among which a notable one is the unanswered question rate. It is frequently observed that in spite of active participation in CQA sites, a significant portion of questions remain unanswered (Dror, Maarek, and Szpektor 2013). This phenomenon widely exists in CQA systems and is often referred as question starvation (Li and King 2010). Several efforts have been made to reduce the number of unanswered questions and one method has been attached much importance in the community, namely how to proactively seek knowledge from historical question/answer pairs.

Normally CQA sites have accumulated large archives of historical questions and associated answers. Retrieving relevant historical question/answer pairs which best match a user's new question or search query is an essential component of a CQA service. Additionally, when a user asks a new question in CQA service and good matches can be located, the lag time incurred by having to wait for a person to respond can be avoided, thus improving user satisfaction.

Question/answer pair retrieval has three major methods: 1) finding historical questions which are similar to the target question. For example, Carmel et al. tried to rank historical questions using both inter-question and question-answer similarity to respond to a newly posed question (Carmel, Shtalhaim, and Soffer 2000) (Figueroa and Neumann 2014); 2) identifying the most relevant answers to the target question within a collection of answers. A representative application was proposed by Surdeanu et al. (Surdeanu, Ciaramita, and Zaragoza 2008), who combine translation and similarity features to recommend historical answers by relevance to a given question, though it focuses only on *how to* questions; 3) combining question similarity and answer relevance. Shtok et al. proposed a two stage approach (Shtok et al. 2012), in the first stage historical questions similar to the new question are identified and ranked so as to produce a single resolved question candidate, in the second stage the best answer to the top candidate question is evaluated in order to verify whether it meets the underlying needs of the new question.

In this paper, we mainly focus on identifying the most relevant answers from a collection of answers by calculating the matching probability between the question q and each candidate answer d. The candidate answers are then ranked by their probability. Intuitively, many bag-of-words based models can be applied to solve this kind of problem, including vector space model (VSM) (Jeon, Croft, and Lee 2005a; 2005b; Duan et al. 2008), language model (LM) (Jeon, Croft, and Lee 2005b; Duan et al. 2008), and Okapi model (Okapi) (Jeon, Croft, and Lee 2005b). These early ap-

Copyright © 2015, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

proaches are normally based purely on lexical matching techniques in the form of exact string matches for n-grams, as such they will fail to detect similar meaning which is conveyed by synonymous words. To overcome this problem, several approaches have been proposed: Wordnet-based and corpus-based semantic similarity measures, the translation model (TM) (Jeon, Croft, and Lee 2005b; Riezler et al. 2007; Xue, Jeon, and Croft 2008), and translation based language model (TRLM) (Cao et al. 2010).

These bag-of-words based schemas, although proven to be effective for tasks like information retrieval, are often incapable of modelling the matching between complicated and structured objects. Firstly, a text contains both syntactical and lexical information, however bag-of-words models normally put the structural information aside, e.g., the word sequence information. As a result, in some worst cases, though two phrases have the same bag-of-words representation, their real meaning is totally opposite (Socher et al. 2011b). Secondly, pre-defined functions cannot sufficiently take into account the complicated interaction between components within the texts (Lu and Li 2013), thereby making it more reasonable to use a trainable model.

In this research we try to make use of the most basic structural information, word sequence, to help improve the question and answer matching precision. Instead of simply disorganizing text into a set of words, we directly model sentences into an ordered list of vectors, where each unique vector represents a unique word. Both sequential and lexical information are stored in the ordered list of word vectors. Afterwards, the complex interaction between questions and answers is modelled into a similarity matrix (S-matrix). Finally, the pattern of the best matching QA pairs is recovered by a deep architecture. It is believed that this approach is able to explicitly capture the word sequence information and the lexical information in matching two structured objects. The experimental study conducted on a dataset from Baidu Zhidao shows promising results.

The rest of the paper is organised as follows. The proposed model will be illustrated in detail in Section 2. Section 3 covers the experimental study and also learned lessons. Section 4 will present related work in solving the CQA matching problem. Section 5 concludes the paper and highlights possible future research directions.

Methodology

In this paper, we aim to solve the problem of matching questions and answers in a CQA system by considering both lexical meaning and word sequence information. The basic architecture of the proposed model is depicted in Fig. 1, where three parts are included and summarised below:

S1 Questions and answers are represented as an ordered list of word vectors using the neural language model;

S2 The complex interaction between questions and answers is modelled into a two-dimensional matrix "S-matrix";

S3 A deep convolutional neural network is trained to give suitable answer probability.

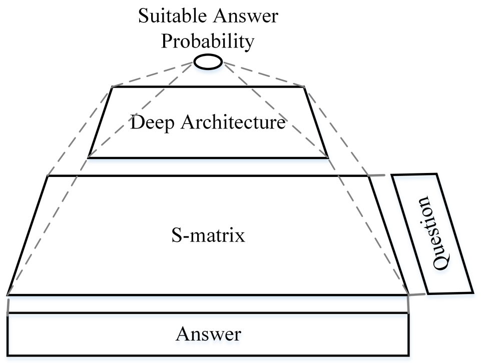

Figure 1: Architecture for Question and Answer Matching Approach

Question/Answer Sentences Representation

The first task of matching questions and answers is to properly present the sentences. In this research we employ the idea of neural language model (Bengio et al. 2006), which is able to jointly learn embedding of words into an n-dimensional vector space and then to use these vectors to predict how likely a word is given its context. One of the popular methods to calculate such embedding information is a model called skip-gram (Mikolov et al. 2013). When skip-gram networks are optimised via gradient ascent, the derivatives will modify the word embedding matrix $L \in R^{(n \times |V|)}$, where $|V|$ is the size of the vocabulary. The word vectors inside the embedding matrix will capture distributional syntactic and semantic information via the word co-occurrence statistics (Bengio et al. 2006; Collobert and Weston 2008; Mikolov et al. 2013).

Once this matrix is learned on an unlabelled corpus, it can be used for subsequent tasks by using each word's vector (a column in L) to represent that word. In the remainder of this paper, we represent a short text as an ordered list of these vectors $(x_1, ..., x_m)$. This representation contains more information than the bag-of-words representation. For example, the sentence "Why don't cat eat mice?" is transformed into a list of vector $(x_{why}, x_{don't}, x_{cat}, x_{eat}, x_{mice})$.

S-matrix

After the sentences in a question/answer pair are properly presented, the next step is to model the complex interaction between questions and answers. We calculate the cosine vector similarity between all words within the two sentences. These similarities fill a matrix as the one shown on the left side of Fig. 2. Consider a pair of question (represented as $(x^q_1, ..., x^q_n)$) and candidate answer (represented as $(x^d_1, ..., x^d_m)$) with the lengths n and m, respectively, it is able to map this pair into a matrix Σ of size $n \times m$, where σ_{ij} is defined as below:

$$\sigma_{ij} = cosine(x^q_i, x^d_j) \quad (1)$$

However, since the matrix dimensions vary according to the sentence lengths, we cannot simply feed the matrix into a standard neural network or classifier. As such in this paper,

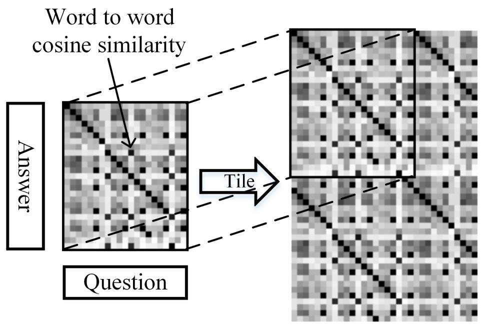

Figure 2: Example of Original Matrix and Tiled Matrix (darker pixel represent higher similarity).

we simply tile the matrix into a larger matrix of fixed size $n_f \times m_f$, as shown in right side of Fig. 2. As a result, it is able to keep all information stored in this matrix during the transformation. The matrix of fixed size is called an S-matrix, denoted by S. Although this approach limits us to deal with question and answer pairs with length smaller than n_f and m_f. In practice, most of the questions and answers in CQA sites are relatively short.

Suitable Answer Probability Estimation

Simply extracting the aggregate statistics of this matrix, such as the average similarity or a histogram of similarities, cannot accurately capture the global structure of the S-matrix. In order to utilise the information stored in the S-matrix to estimate whether the answer is suitable for the question, a deep convolution network (DCNN) (LeCun et al. 1998) is employed to create a function $f(S)$:

$$p_{d|q} = f(S) \tag{2}$$

where $p_{d|q}$ refers to the possibility of the answer d being suitable for the question q.

DCNN are variants of MLPs which are inspired from biology. From Hubel and Wiesel's early work on the cat's visual cortex (Hubel and Wiesel 1962), it is known that there exists a complex arrangement of cells within the visual cortex. These cells are sensitive to small sub-regions of the input space, called receptive fields, and are tiled in such a way as to cover the entire visual field. These filters are local in input space and are thus better suited to exploit the strong spatially local correlation present in natural images[1].

DCNN is employed in this research because the pattern, which signifies whether question and answer are matched, can be found in any part of the original S-matrix. Firstly, the original matrix is tiled into a bigger matrix, thus the same pattern can be repeated many times in the bigger matrix. Secondly, the question can be answered in different parts of the answer body, i.e., head, middle, or end.

The overall structure is illustrated in Fig. 3. The first convolution layer C1 is used to estimate phrase-level match score between the question and the answer. The input hidden units in C1 are connected to a local sub-matrix in the input S. If the k-th feature map at layer C1 is denoted as h^k, whose filters are determined by the weights W^k and bias b_k, x_{ij} denote a sub-matrix of S-matrix, then the feature map h^k is obtained as follows (for tanh non-linearity):

$$h^k_{ij} = tanh((W^k * x_{ij}) + b_k) \tag{3}$$

The matching score for each sub-matrix in S is calculated but only part of these sub-matrices contain useful information. The first max-pooling layer M1 is used to eliminate useless match scores. Max-pooling partitions the input matrix into a set of non-overlapping rectangles and for each such sub-region it outputs the maximum value. Layer C2 is used to estimate sentence-level match score and the input hidden units are connected to a local subset of units in M1. the second max-pooling layer M2 eliminates useless match scores in C2. Finally, a fully connected layer is used to estimate the final match score. Fig. 4 gives us some insights into layer C1 and M1. Convolutional layers C1 and C2 both contain multiple feature maps. Because similarity between phrases and sentences can be evaluated in variety of perspectives (Mikolov et al. 2013), our experiments will further confirm the usefulness of multiple feature maps.

We employ a discriminative training strategy with a large margin objective. Suppose that we are given the following triples (x, y^+, y^-) from the oracle, with x $(\in X)$ matched with y^+ better than with y^- (both $\in Y$). We have the following ranking-based loss as objective:

$$L(W, D_{trn}) = \sum_{(x_i, y_i^+, y_i^-) \in D_{trn}} e_W(x_i, y_i^+, y_i^-) \tag{4}$$

where $e_W(x_i, y_i^+, y_i^-)$ is the error for triple (x_i, y_i^+, y_i^-), given by the following large margin form:

$$\begin{aligned} e_i &= e_W(x_i, y_i^+, y_i^-) \\ &= max(0, m_a + s(x_i, y_i^-) - s(x_i, y_i^+)) \end{aligned} \tag{5}$$

with $0 < m_a < 1$ controlling the margin in training.

Experimental Study

Dataset

For the purpose of evaluating and validating the proposed model, a dataset is installed by collecting data from Baidu Zhidao[2], the biggest Chinese CQA site. We randomly collected 438,078 questions throughout March 2013 from this CQA site. Among these questions, 303,588 (69.3%) questions have at least one answer, 196,879 (44.9%) questions have been solved (at least one answer has been marked as the best answer), while 134,490 (30.7%) questions have not received even one answer after 6 months since posting. This finding of question starvation is in accordance with previous report (Li and King 2010).

From this dataset, 196,879 solved questions are employed to create training data and testing data. Since S-matrix limits the maximum length of questions and answers to n_f and m_f, it is necessary to firstly select proper n_f and m_f values. Table 1 shows the number and proportion of questions

[1]http://deeplearning.net/tutorial/lenet.html

[2]http://zhidao.baidu.com/

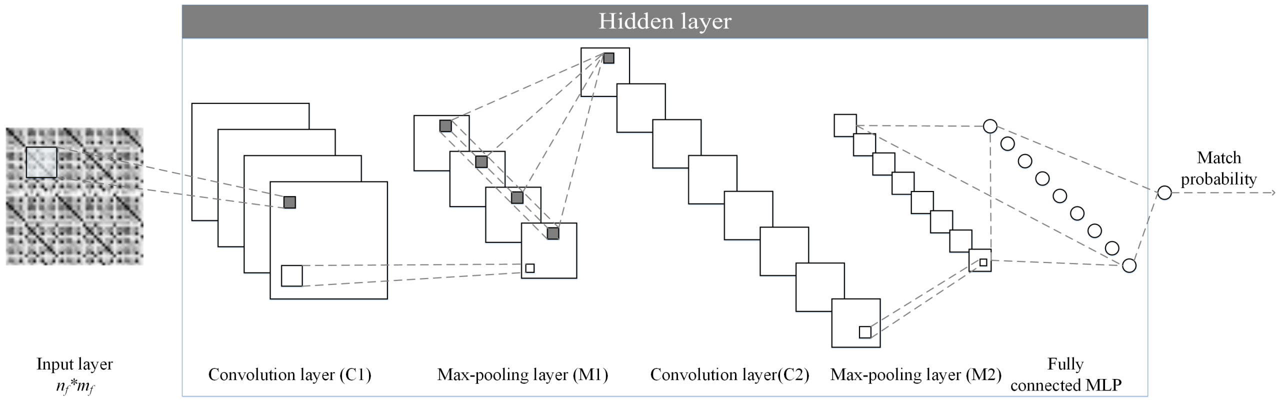

Figure 3: Architecture of Deep Convolutional Neural Network

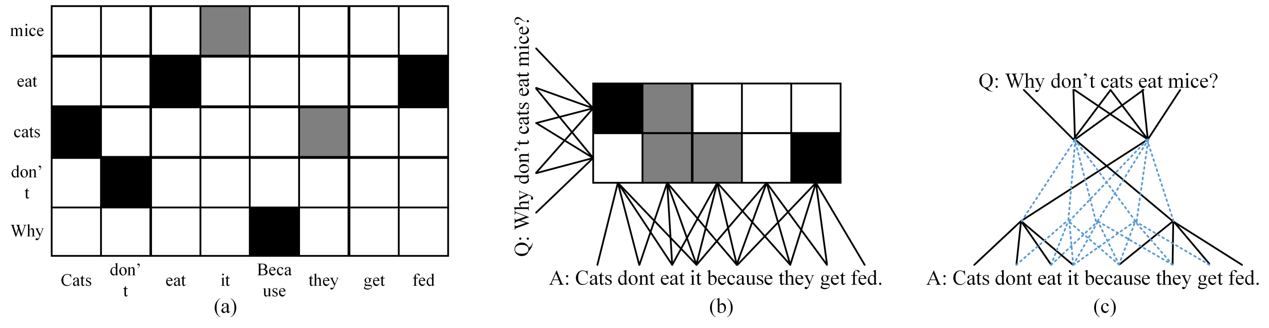
mice
eat
cats
don't
Why
Cats
don't
eat
it
Because
they
get
fed
(a)
Q: Why don't cats eat mice?
A: Cats dont eat it because they get fed.
(b)
Q: Why don't cats eat mice?
A: Cats dont eat it because they get fed.
(c)

Figure 4: (a) is S-matrix(S) generated from question "Why don't cat eat mice?" and answer "Cats don't eat it because they get fed." (darker grid represents higher similarity or matching score) (b) is layer C1 result of S. In this example, units in C1 are connected to a $4 * 4$ local sub-matrix of S, weight W^k and bias b_k are trained on large QA archive. Layer M1 filters out low similarity grids, keeps high similarity dark grids. (c) emphasises the ability of matching question and answer at phrase level. With convolution layer, n-gram matching score is calculated. With pooling layer, important relation can be identified (solid line in (c)).

Table 1: Number and Coverage proportion of questions with different n_f and m_f

n_f	m_f	Question#	Coverage proportion
30	50	99,909	50.7%
40	60	115,666	58.7%

covered by different n_f and m_f. From the table it is observed that lengths of more than half of question and answer pairs are less than 30 and 50, respectively. Furthermore, if we gradually increase n_f and m_f from $(30, 50)$ to $(40, 60)$, the coverage only increases by 8.0%, but the cost of memory increases by 60%. Thus, it is reasonable to choose $n_f = 30$ and $m_f = 50$ in our experimental study. In real application, this model can be trained with high performance computing platform or parallel training method, n_f and m_f can be increased to several hundreds. As such, much more questions and answers can be covered.

To better evaluate the proposed method, several data subsets are utilised. Firstly, all 99,909 questions are randomly split into training and testing data. Training data contains 90,000 questions and 90,000 (x, y^+, y^-) triples are generated, where x is a solved question, y^+ is the best answer, and y^- is a randomly selected answer which belongs to a random category. Secondly, another 4 groups of 90,000 (x, y^+, y^-) triples are generated. They are noted as training data Tr_2, Tr_3, Tr_4, and Tr_5. Training data Tr_2 contains 18,000 different questions which belong to the category 'Computers & Internet'. Each pair of x and y^+ appear in 5 different triples with different y^-. Consequently, 90,000 different triples are generated. Training data Tr_3, Tr_4, and Tr_5 are produced with same method using questions in "Education & Science", "Games", and "Entertainment & Recreation". The remaining questions in these categories make up testing data Te_2, Te_3, Te_4, and Te_5. Each testing data contain at least 1,000 different questions. All models' parameters are learned from the training data. The hyper parameters are tuned on a validation set (as part of the training set).

Evaluation Metrics & Baseline

In order to evaluate the accuracy of matching questions and answers, a set of candidate answers is created with size 6 (one positive + five negative) for each question in the testing data. In Te_1 negative candidates usually belong to different categories to the questions, while in Te_{2-5} negative candidates always belong to the same category as the question. We compare the performance of our approach in ranking quality of the six candidate answers against that of others baselines. Discounted cumulative gain (DCG) (Järvelin and Kekäläinen 2000) is employed to evaluate the ranking quality. The premise of DCG is that highly relevant documents appearing lower in a ranking list should be penalised as the graded relevance value is reduced logarithmically proportional to the position of the result. DCG accumulated at a particular rank position p is defined as:

$$\text{DCG@p} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i)} \tag{6}$$

where best answer $rel = 1$, for other answers $rel = 0$. We choose DCG@1 to evaluate the precision of first rank result and DCG@6 to evaluate the quality of ranking, similar to the metrics introduced in (Lu and Li 2013).

We compare our S-matrix + DCNN approach against five popular retrieval models: Okapi model (Okapi) (Jeon, Croft, and Lee 2005b), Language model (LM) (Jeon, Croft, and Lee 2005b; Duan et al. 2008), Translation model (TM) (Jeon, Croft, and Lee 2005b; Riezler et al. 2007; Xue, Jeon, and Croft 2008), Translation based language model (TRLM) (Cao et al. 2010) and Random guess. The model parameters setup are same as in (Cao et al. 2010), but we use word vector cosine similarity for word translation probabilities in TM and TRLM model.

Experiment Settings

After the choice of n_f and m_f, each (x, y^+) and (x, y^-) pair becomes a 30 × 50 matrix and the training set is then transformed into 180,000 matrixes. Our goal is to train a classifier which can tell whether one matrix is formed by (x, y^+) or (x, y^-). In the following, convolution layers are labelled Cx, and max-pooling layers are labelled Px, where x is the layer index. Each unit in convolutional layer is connected to a 5 × 5 neighbourhood in the previous layer. Each unit in max-pooling layer Px is connected to a 2 × 2 neighbourhood in the corresponding feature map in Cx. Layer C1 is a convolutional layer with 20 feature maps. Layer M1 is a max-pooling layer with 20 feature maps. Layer C2 is a convolutional layer with 50 feature maps. Layer M2 is a max-pooling layer with 50 feature maps. Fully connected layer contains 500 units and is fully connected to M2. It has 900,000 trainable parameters in total.

Fig. 5 shows the evolution of nDCG@1 and nDCG@6 with different margin m_a. It is observed that 0.2 is the best choice of m_a. Table 2 shows that multiple feature maps in C1 and C2 achieve better precision than uni-feature map.

Result and Discussion

From Table 3, it is observed that S-matrix + DCNN, TRLM and TM perform better than the other methods on Te_1. This is because word embedding can capture more lexical information than traditional approaches. When comparing S-matrix + DCNN against the TRLM model, we are able to see that the improvement is due to extra structural information.

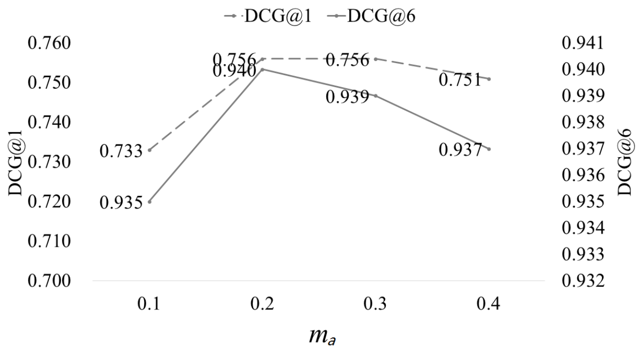

Figure 5: Evolution of DCG@1 and DCG@6 while m increasing

Table 2: Performance difference between uni-feature map and multiple feature map

Feature map in C1 & C2	1-1	20-50
DCG@1	0.691	**0.752**
DCG@6	0.927	**0.943**

Table 3: Performance of different approach on random category training data Te_1

Approach	DCG@1	DCG@6
S-matrix + DCNN (trained on Tr_1)	**0.752**	**0.943**
TRLM	0.696	0.924
TM	0.690	0.922
LM	0.634	0.819
Okapi	0.608	0.807
Random guess	0.167	0.550

From Table 4, it is observed that if the questions and answers are limited to a specific category, accuracy will decrease to some extent. This is because, within a single category, inappropriate answers are more similar to the best answer than random selected answers. In those cases, S-matrix + DCNN can still achieve better performance than traditional approaches. To further analyse the performance, we compared DCNN with Matrix sum (add all similarity together), Multilayer Perceptron (MLP) with a single hidden layer, and Logistic regression. Table 5 shows that DCNN can achieve a better performance than other methods. As such we conclude that DCNN has the ability to discover hidden information in S-matrix due to its ability to exploit the strong spatially local correlations.

Related work

Retrieving relevant historical answers in a large archive of questions and associated answers is an essential function of a CQA site. Most existing works employ bag-of-words based schema. However, in the question/answer matching process both syntactic and lexical information are important, but bag-of-words models remove the order of words. As a

Table 4: Performance of different approaches on mono-category training data Tr_{2-5}, S+DCNN (Tr_i) represents the training data belonging to the same category as the testing data

Approach	Te_2		Te_3		Te_4		Te_5	
	DCG@1	DCG@6	DCG@1	DCG@6	DCG@1	DCG@6	DCG@1	DCG@6
S+DCNN (Tr_i)	**0.658**	**0.912**	0.734	**0.939**	**0.619**	**0.894**	**0.543**	**0.866**
TRLM	0.601	0.885	0.698	0.924	0.562	0.865	0.492	0.843
TM	0.596	0.885	0.691	0.922	0.560	0.863	0.486	0.841
LM	0.624	0.830	**0.746**	0.881	0.544	0.765	0.488	0.740
Okapi	0.567	0.806	0.702	0.869	0.467	0.747	0.446	0.723
Random guess	0.167	0.550	0.167	0.550	0.167	0.550	0.167	0.550

Table 5: Performance of different approaches with the same S-matrix on Te_1

Approach	DCG@1	DCG@6
S-matrix + DCNN (trained on Tr_1)	**0.752**	**0.943**
S-matrix + Matrix sum	0.676	0.929
S-matrix + MLP	0.671	0.926
S-matrix + logistic	0.652	0.920
Random guess	0.167	0.550

result two irrelevant phrases could probably have the same bag-of-words representation (Socher et al. 2011b). To solve this problem, several approaches have been proposed. A solution in (Duan et al. 2008) considers the question structure for retrieval by building a structured tree for questions. Similarly, Bian et al. proposed an interesting learning framework for question retrieval (Bian et al. 2008). However, Wang et al. observed that current parsers are not well-trained for real-life questions, especially in informally stated questions (Wang, Ming, and Chua 2009). These proposed approaches need training data (that are difficult to get for general questions) and the experimental studies are conducted on factoid questions. Inspired by technologies in the image processing field, our work captures both lexical and sequential information in matching process. We focus on matching questions and answers, therefore a large scale of training data can be constructed based on the past question and answer pairs.

Our work is conceptually similar to the dynamic pooling algorithm for paraphrase identification, recently developed by Socher et al. (Socher et al. 2011a). Similar to our proposed model, their model constructs a neural network on the interaction space of two objects (sentences in their case), and outputs the measure of semantic similarity between them. The major differences are two-fold: 1) their model utilises parse trees to capture the structure information of sentences, and finds vector representations for each node of a parse tree using recursive autoencoders; 2) a dynamic pooling model is used to fit the similarity matrix into a fixed shape. Instead, our model employs two convolutional layers to capture the phrase and sentence level structural information, and tile the similarity matrix into a larger matrix: this is a process without loss of information.

Our work is in a sense related to that presented in (Lu and Li 2013). This work directly models object-object interactions with a deep architecture. This architecture is able to explicitly capture the natural non-linearity and the hierarchical structure in matching two structured objects. The main difference between their approach and our approach is that this model is also based on a bag-of-words schema. Their model represents each word with a single value. We use a 100-dimensional word embedding vector to represent words, and multiple senses of similarity between sentences are also considered through multiple feature maps.

Our work is also conceptually related to the convolutional neural network for modelling sentences (Kalchbrenner, Grefenstette, and Blunsom 2014). The idea of using DCNN to model phrase and sentence level property is similar. Different from our work, this work uses DCNN to model single sentence.

Conclusion and Future Works

In this research we presented a novel approach to meet the short text matching problems in CQA applications, inspired partially by the long thread of work on deep learning. The major contribution of this work consists of two parts. Firstly, questions and answers are modelled with an S-matrix by considering both lexical and structural interaction. Secondly, an effective approach of matching question and answer is realised by using a deep convolutional neural network. The experiments have shown the potential of the proposed method as compared against other popular methods. It is believed that this work might offer an insight for information seeking in CQA sites.

Our work opens several interesting new directions for future work. It is possible, for example, to try other word to word similarities, and the similarity between words can be also expressed as a vector as in (Mikolov et al. 2013). It is also interesting to explore the possibility of using this approach to solve other short text matching tasks (e.g., evaluation of machine translation, comment selection on a given tweet): this deserves further investigation in future work.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (No. 61332018), the National High Technology Research and Development Program of China (No. 2013AA01A601), and the Fundamental Research Funds for the Central Universities. We are grateful to Shenzhen Key Laboratory of Data Vitalization (Smart City) for supporting this research.

References

Bengio, Y.; Schwenk, H.; Senécal, J.-S.; Morin, F.; and Gauvain, J.-L. 2006. Neural probabilistic language models. In *Innovations in Machine Learning*. Springer. 137–186.

Bian, J.; Liu, Y.; Agichtein, E.; and Zha, H. 2008. Finding the right facts in the crowd: factoid question answering over social media. In *Proceedings of the 17th International Conference on World Wide Web*, 467–476.

Bilotti, M. W.; Elsas, J. L.; Carbonell, J. G.; and Nyberg, E. 2010. Rank learning for factoid question answering with linguistic and semantic constraints. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management*, 459–468.

Cao, X.; Cong, G.; Cui, B.; and Jensen, C. S. 2010. A generalized framework of exploring category information for question retrieval in community question answer archives. In *Proceedings of the 19th International Conference on World Wide Web*, 201–210.

Carmel, D.; Shtalhaim, M.; and Soffer, A. 2000. eresponder: Electronic question responder. In *Proceedings of the 7th International Conference on Cooperative Information Systems*, 150–161.

Chua, A. Y. K., and Banerjee, S. 2013. So fast so good: An analysis of answer quality and answer speed in community question-answering sites. *Journal of the American Society for Information Science and Technology* 64(10):2058–2068.

Collobert, R., and Weston, J. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, 160–167.

Dror, G.; Maarek, Y.; and Szpektor, I. 2013. Will my question be answered? predicting "question answerability" in community question-answering sites. In *Proceedings of 2013 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, Part III*, 499–514.

Duan, H.; Cao, Y.; Lin, C.; and Yu, Y. 2008. Searching questions by identifying question topic and question focus. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, 156–164.

Figueroa, A., and Neumann, G. 2014. Category-specific models for ranking effective paraphrases in community question answering. *Expert Systems with Applications* 41(10):4730–4742.

Hubel, D. H., and Wiesel, T. N. 1962. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology* 160(1):106–154.

Järvelin, K., and Kekäläinen, J. 2000. IR evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 41–48.

Jeon, J.; Croft, W. B.; and Lee, J. H. 2005a. Finding semantically similar questions based on their answers. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 617–618.

Jeon, J.; Croft, W. B.; and Lee, J. H. 2005b. Finding similar questions in large question and answer archives. In *Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management*, 84–90.

Kalchbrenner, N.; Grefenstette, E.; and Blunsom, P. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 655–665.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

Li, B., and King, I. 2010. Routing questions to appropriate answerers in community question answering services. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management*, 1585–1588.

Lu, Z., and Li, H. 2013. A deep architecture for matching short texts. In *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, 1367–1375.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *CoRR* abs/1301.3781.

Riezler, S.; Vasserman, A.; Tsochantaridis, I.; Mittal, V. O.; and Liu, Y. 2007. Statistical machine translation for query expansion in answer retrieval. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*.

Shtok, A.; Dror, G.; Maarek, Y.; and Szpektor, I. 2012. Learning from the past: answering new questions with past answers. In *Proceedings of the 21st International Conference on World Wide Web*, 759–768.

Socher, R.; Huang, E. H.; Pennington, J.; Ng, A. Y.; and Manning, C. D. 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems*, 801–809.

Socher, R.; Pennington, J.; Huang, E. H.; Ng, A. Y.; and Manning, C. D. 2011b. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 151–161.

Surdeanu, M.; Ciaramita, M.; and Zaragoza, H. 2008. Learning to rank answers on large online QA collections. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, 719–727.

Wang, K.; Ming, Z.; and Chua, T. 2009. A syntactic tree matching approach to finding similar questions in community-based QA services. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 187–194.

Xue, X.; Jeon, J.; and Croft, W. B. 2008. Retrieval models for question and answer archives. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 475–482.