arXiv:1707.05409v1 [cs.IR] 17 Jul 2017

Neural Matching Models for Question Retrieval and Next Question Prediction in Conversation

Liu Yang[1] Hamed Zamani[1] Yongfeng Zhang[1] Jiafeng Guo[2] W. Bruce Croft[1]
[1] Center for Intelligent Information Retrieval, University of Massachusetts Amherst, Amherst, MA, USA
[2] CAS Key Lab of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
{lyang,zamani,yongfeng,croft}@cs.umass.edu,guojiafeng@ict.ac.cn

ABSTRACT

The recent boom of AI has seen the emergence of many human-computer conversation systems such as Google Assistant, Microsoft Cortana, Amazon Echo and Apple Siri. We introduce and formalize the task of predicting questions in conversations, where the goal is to predict the new question that the user will ask, given the past conversational context. This task can be modeled as a "sequence matching" problem, where two sequences are given and the aim is to learn a model that maps any pair of sequences to a matching probability. Neural matching models, which adopt deep neural networks to learn sequence representations and matching scores, have attracted immense research interests of information retrieval and natural language processing communities. In this paper, we first study neural matching models for the question retrieval task that has been widely explored in the literature, whereas the effectiveness of neural models for this task is relatively unstudied. We further evaluate the neural matching models in the next question prediction task in conversations. We have used the publicly available Quora data and Ubuntu chat logs in our experiments. Our evaluations investigate the potential of neural matching models with representation learning for question retrieval and next question prediction in conversations. Experimental results show that neural matching models perform well for both tasks.

KEYWORDS

Deep learning, neural conversational models, question retrieval, neural networks

1 INTRODUCTION

Due to the ability of neural network models to go beyond term matching similarities as well as omitting the feature engineering steps, neural matching models have recently achieved state-of-the-art performance in a number of information retrieval tasks. However, the generality of these models to be applied on different tasks is relatively unstudied.

In this paper, we focus on two question ranking tasks. The first one is *question retrieval*: retrieving similar questions in response to a specific question. This task is useful in question answering and community question answering (CQA) applications. For instance, finding similar questions could help to improve the question answering accuracy or can help to avoid asking duplicate questions in CQA websites. Although neural approaches have been widely applied to answer sentence selection [6, 21, 26] and similar question identification [30], the effectiveness of deep learning architectures for question retrieval is relatively unstudied. Therefore, we study a set of neural networks that can retrieve similar questions to a given question.

The second task is relevant to conversation models. Building intelligent systems that could perform meaningful conversations

Table 1: Motivated examples of predicting questions in conversations and search. Ground truth labels are highlighted by different text colors, where blue means correct predictions and red means wrong predictions.

Predicting Questions in Conversations as in Ubuntu IRC Chat Rooms
Example 1: Time: 2010-12-18 Conversation Context: [17:23] <neohunter111> Hello I have a problem with my mouse, is a microsoft wireless mouse 7000, when i press button6 or buttton 7 ubuntu recives a lot of press and realease events!! any ideas of how to solve this or how to search in google?? [17:24] <pksadiq> neohunter111: does system > preferences > mouse has any option? [17:26] <neohunter111> pksadiq yes the mouse works, the problem is that i set the boutton 6 and 7 (muse wheel to left o right) to change the desktop screen. and when i press it the desktop cube turns like crazy a lot of times, but before was working ok. [17:27] <pksadiq> neohunter111: go to compiz settings in system > preferences,a dn select 3D desktop plugin and change settings Predicted Question: *Where is 3d desktop plugin?* (Correct) *Is there a keyboard shortcut to change desktop?* (Wrong)
Example 2: Time: 2011-12-22 Conversation Context: [15:59] <gplikespie> Hello, I am new to Linux and am not sure how to move files from windows to linux, can anyone help? [16:02] <etroshica> gplikespie, there is a variety of methods, depending on the file size and how much you want to learn. You can use some basic tools like gmail to Dropbox to send files. If it's a VM you can use shared directories. You can also set up a samba share. If you have ssh access, I recommend winscp, definitely one of the easiest tools to use. Predicted Question: *VM is virtual machine, right?* (Correct) *Would there be any reason why I should use a 32 bit version of Ubuntu instead of 64 bit for a VM?* (Wrong)
Predicting Questions in Search (Question Retrieval) as in Quora
Example 1: Query Question: How can I learn Deep Learning quickly? Predicted Questions: *What are the best resources to learn about deep learning?* (Correct) *How do I learn deep learning in 2 months ?* (Correct) *How is deep learning used in search engines?* (Wrong)
Example 2: Query Question: What made Steve Jobs a great presenter? Predicted Questions: *How can I make a presentation attractively just like Steve Jobs?* (Correct) *What are the secrets behind Steve Jobs' excellent live product presentations?* (Correct) *What was it like to deliver a presentation to Steve Jobs?* (Wrong)

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
SIGIR 2017 Workshop on Neural Information Retrieval (Neu-IR'17), August 7–11, 2017, Shinjuku, Tokyo, Japan
© 2017 Copyright held by the owner/author(s). 123-4567-24-567/08/06.
DOI: 10.475/123_4

with humans has been one of the long term goals of artificial intelligence. Human-computer conversation plays a critical role in many popular mobile search systems, intelligent assistants, and chat bot systems such as Google Assistant, Microsoft Cortana, Amazon Echo, and Apple Siri. Traditional conversational systems are based on hand designed logics and features with natural language templates, which usually only works for restricted and predictable conversational inputs [12, 18, 35]. With rich big data resources on the Web, enhanced GPU computational infrastructures, and large amount of labels derived from crowd sourcing and online user behaviors, end-to-end deep learning methods have begun to show promising results on conversation response ranking and generation tasks [1, 13, 14, 22, 23, 32]. According to these motivations, we focus on a new type of conversational response ranking problem as the second task in the paper: *predicting the next question in a conversation*. During real conversations, humans could not only generate reasonable responses, but also have the ability to predict what the new questions that other speakers will be likely to ask. Learning models that could predict questions in conversations could enable us to better understand user intents during the conversations. Proactive content recommendations could be made without implicit questions issued by users. Furthermore, pre-selected answer sets could be generated based on question prediction results as a cache mechanism to improve the efficiency and effectiveness of conversational question answering systems. Table 1 shows a number of motivated examples of predicting questions in conversations.

Our neural network architecture for both tasks is inspired by previous work [6, 21, 26, 29, 32] that achieves impressive performance in different tasks. The designed siamese neural network models the long dependency of terms using a long short term memory (LSTM) layer. It further takes advantage of multiple convolutional and max pooling layers for representation learning of sequences based on the output of the LSTM layer. The network outputs a real-valued score for each candidate question and all candidate questions are ranked based on their matching score computed by the network.

We evaluate our models for the question retrieval task using the recently released Quora dataset. Our experiments demonstrate that the proposed neural network model outperforms state-of-the-art non-neural question retrieval approaches. The experiments also validate the hypothesis that neural matching models can complement exact term matching approaches in the question retrieval task; hence, a combination of the two is more appropriate. For the next question prediction task, we trained our model on the chat logs extracted from Ubuntu-related chat rooms on the Freenode Internet Relay Chat (IRC) network[1]. Our experiments suggest that neural matching models could perform well for both tasks, which demonstrates the potential of neural matching models with representation learning for new applications and scenarios.

The contributions of this work are as follows: (1) We introduce and formalize the new task of next question prediction in conversations. (2) We study of the effectiveness of neural matching models for question retrieval and predicting questions in conversations. Experimental results show that neural matching models perform well for both tasks.

[1]http://dataset.cs.mcgill.ca/ubuntu-corpus-1.0/

2 NEURAL MATCHING MODEL

2.1 Problem Definition and Model Overview

In this section, we formally explain the high-level architecture of our model. Let matrices $Q \in \mathbb{R}^{l \times |Q|}$ and $P \in \mathbb{R}^{l \times |P|}$ denote the word embeddings of two sequences Q and P, respectively. Let $|\cdot|$ denote the sequence length and l denote the embedding dimensionality of individual vocabulary terms. Each column of Q and P is a word embedding vector representing a word in the sequences. Each sequence pair Q and P is associated with a label y. Given a query sequence Q and multiple candidate sequences $\{P_1, P_2 \cdots P_n\}$, the goal is to generate a candidate sequence rank list. For the question retrieval task, the query sequence Q is a question and the candidate sequences P are the candidate similar questions. For the question prediction task in conversations, the query sequence Q is the previous context, consisting of previous questions with their responses, and the candidate sequences are the next candidate questions.

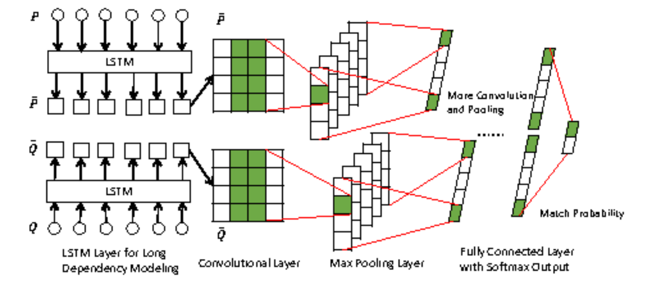

Figure 1: The architecture of LSTM-CNN-Match model for matching sequences.

Figure 1 shows the architecture of LSTM-CNN-Match model for matching sequences. This model is an extension of the CDNN model proposed by Severyn and Moschitti [21] that has been also explored in various applications such as answer sentence selection [6, 26, 29, 32, 33, 39]. Comparing with CDNN, this model adopts a long short term memory (LSTM) layer for long term dependency modeling in sequences. The convolutional layers are running on the output of the latent representations modeled by the LSTM layer, instead of the raw word embeddings sequence. In the following, we describe the model in more detail.

2.2 LSTM for Long Term Dependency Modeling

We use an LSTM layer to process Q and P for modeling long term dependency information in the sentences. LSTM [7] is an advanced variant of recurrent neural networks (RNN). It can overcome the vanishing / exploding gradient problem of simpler Vanilla RNNs with the memory cell and gating mechanisms. Each LSTM cell consists of a memory cell that stores information over a long history and three gates that specify how to control the information flow into and out of the memory cell. Given an input sequence $Q = (x_0, x_1, ..., x_t)$, where x_t denotes the word embedding at position t, LSTM outputs a new representation matrix $\bar{Q}$ that captures contextual information seen before in addition to the word at position t itself based on the equations below:

$$\begin{aligned}
i_t &= \delta(W^i x_t + U^i h_{t-1} + b^i) && (1)\\
f_t &= \delta(W^f x_t + U^f h_{t-1} + b^f) && (2)\\
o_t &= \delta(W^o x_t + U^o h_{t-1} + b^o) && (3)\\
u_t &= \tanh(W^u x_t + U^u h_{t-1} + b^u) && (4)\\
c_t &= i_t u_t + f_t c_{t-1} && (5)\\
h_t &= o_t \tanh(c_t) && (6)
\end{aligned}$$

where i, f, o denote the input, forget and output gates, respectively. c is the stored information in the memory cells and h is the learned representation. Thus h_t is corresponding to the t-th column of the new representation matrix $\bar{\mathbf{Q}}$ which encodes the t-th word in $\mathbf{Q}$ with its context information. We also tried to use the bidirectional LSTM (Bi-LSTM). But we found that Bi-LSTM does not improve the performance. It led to lower training efficiency comparing with LSTM. Thus we just use one directional LSTM in our model.

2.3 Convolutional and Max Pooling Layers

Given the hidden representations learned by the LSTM layer, we use convolutional layers with different filter sizes and max pooling layers with different window sizes to learn sequence representations for generating the matching score. The convolution operation transforms the original feature map to a new feature map by moving the filters and computing the dot products of the filters with the corresponding feature map patch. Each filter slides over the whole embedding vectors, but varies in how many words it covers.[2] We slide the filters without padding the edges and perform a narrow convolution [10] . We further feed the output of the convolutional layer to a rectified linear unit (ReLU) function which is simply defined as $\max(0, \mathbf{x})$ to add non-linearity. After that we apply a max pooling layer on the output of the ReLU function. Finally we use a fully connected layer with a softmax function to output the probability distribution over different labels.

2.4 Loss Function and Training

We consider a pairwise learning setting during model training process. The training data consists of triples $(\mathbf{Q}_i, \mathbf{P}_i^+, \mathbf{P}_i^-)$ where $\mathbf{P}_i^+$ and $\mathbf{P}_i^-$ respectively denote the positive and the negative candidate sequence for $\mathbf{Q}_i$. The pairwise ranking-based hinge loss function is defined as:

$$\mathcal{L} = \sum_{i=1}^{M} \max(0, \epsilon - S(\mathbf{Q}_i, \mathbf{P}_i^+) + S(\mathbf{Q}_i, \mathbf{P}_i^-)) + \lambda||\theta||_2^2 \qquad (7)$$

where M is the number of triples in the training data. $\lambda||\theta||_2^2$ is the regularization term where λ and θ respectively denote the regularization coefficient and the model parameters. ϵ denotes the margin in the hinge loss. $S(\cdot,\cdot)$ denotes the output matching score from the last layer of the LSTM-CNN-Match model. The parameters of the network are optimized using the *Adam* algorithm [11].

[2]We set filter sizes to [1, 2, 3, 4, 5] and use 128 filters of each size in our model.

3 EXPERIMENTS

3.1 Datasets and Experimental Setting

We use the publicly available datasets from Quora[3] and Ubuntu IRC chat logs[4] for the experiments. The Quora dataset consists of 404, 340 lines of question pairs. Each line contains the IDs for each question in the pair, the full text for each question, and a binary value that indicates whether the line contains a similar question pair or not. To use this dataset for question retrieval evaluation, we conducted data sampling and pre-processing. There are 148, 487 similar question pairs in the Quora data, which form the positive question pairs. For each positive question pair, we randomly picked one of them as the query question $\mathbf{Q}$. Then the other question is the positive candidate question $\mathbf{P}^+$ for $\mathbf{Q}$. We used negative sampling to construct the negative pairs following previous work [27]. Specifically for each query question $\mathbf{Q}$, we first used it to retrieve the top 1000 results from the whole question set using Lucene[5] with BM25. Then we randomly selected 4 questions from them except the known positive candidate question $\mathbf{P}^+$ to construct the negative candidate questions. Finally, we randomly separated the whole dataset to training, development and testing data with proportion 8 : 1 : 1. The statistics of different data partitions of the Quora data is presented in Table 2.[6]

For the Ubuntu chat log data, we also perform similar data sampling and pre-processing. We identify questions from dialogs by question marks. For each question q^* in a dialog, we stochastically sample a pre-context size $c \in [2, C]$, where C is the max number of questions in the pre-context.[7] Let $c' = \min(c, t)$, where t is the total number of questions before q^*. Then we generate context for q^* by merging previous c' questions $\{q_1, q_2, \cdots, q_{c'}\}$ with their responses. Thus the true question response q^* is the positive question candidate. We additionally randomly sample another 9 negative question responses except the known positive candidate question following previous work [15]. Finally, we randomly separated the whole dataset to training, development and testing data with proportion 8 : 1 : 1. The statistics of different data partitions of the Ubuntu chat log data is presented in Table 3.

For data pre-processing, we performed tokenization and punctuation removal. We maintained stopwords for neural models and removed them for the traditional retrieval models such as BM25 and QL. We used TensorFlow[8] for the implementation of the neural matching models.

Word Embeddings. We use Glove [19] word embeddings, which are 300-dimension word vectors trained with a crawled large corpus with 840 billion tokens. Embeddings for words not present are randomly initialized with sampled numbers from a uniform distribution U[-0.25,0.25], which follows the same setting as in [21].

[3]https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs
[4]http://dataset.cs.mcgill.ca/ubuntu-corpus-1.0/
[5]http://lucene.apache.org/
[6]Note that in some rare cases, the hits count for a query question returned by Lucene could be less than 4. In this case, the actual candidate question number for this query question could be less than 5.
[7]In our experiments, we empirically set $C = 6$. We skip a question if there are less than 2 previous questions or the question length is less than 3. We remove speaker IDs in candidate questions to insure that different methods rank questions by matching actual question content instead of spearker IDs. Words appear less than or equal to 5 times are replaced by <UNK>.
[8]https://www.tensorflow.org/

Table 2: The statistics of Quora data.

Data	Train	Dev	Test	Total
#QueryQ	118,789	14,848	14,850	148,487
#CandidateQ	593,932	74,240	74,250	742,422
AvgQueryQLen	9.81	9.88	9.87	9.85
AvgCandidateQLen	9.91	9.89	9.92	9.91

Table 3: The statistics of Ubuntu chat log data.

Data	Train	Dev	Test	Total
#Context	102,680	12,994	12,896	128,570
#CandidateQ	1,026,800	129,940	128,960	1,285,700
AvgContextLen	125.85	125.40	125.44	125.76
AvgCandidateQLen	14.59	14.56	14.55	14.59

Additional Word Overlap Features. As noted in previous work [21, 36], one weakness of models relying on distributional word embeddings is their inability to deal with cardinal numbers and proper nouns. This also has impacts on matching question pairs or contexts with questions. Suppose we have two questions "*What happened in US in 1776?*" and "*What happened in Japan in 1871?*". These two questions will be likely predicted with a high matching probabilities by neural matching models replying on word embedding input since country names like "US" and "Japan", numbers like "1776" and "1871" have close distances in the word embedding space. However, these two questions represent two different question intents. To mitigate this issue, we follow the approach in [21, 36] and include additional word overlap features into the model. Specifically, we compute the word co-occurrence count and IDF weighted word co-occurrence between two sequences. Computing these simple word overlap features is straightforward. We combine the matching probability learned by neural matching models with these two simple word overlap features with a logistic regression layer to generate the final ranking scores of candidate questions.

Model Hyper-parameters. We tuned the hyper-parameters with grid search using the development set. For the setting of LSTM-CNN-Match model in question retrieval, we set learning rate to 0.002, batch size to 500, margin of the hinge loss to 0.5, filter sizes to $[1, 2, 3, 4, 5]$, and the number of each feature size to 128. For the setting of LSTM-CNN-Match model in question prediction in conversations, we set learning rate to 0.002, batch size to 200, margin in the hinge loss to 0.3, filter sizes to $[1, 2, 3, 4, 5]$, and the number of each feature size to 128.

3.2 Evaluation Metrics and Compared Methods

For the Quora data and Ubuntu chat log data, since there is only one positive candidate question for each query question or previous conversation context, we adopt mean reciprocal rank (MRR) and precision at the highest position (P@1) as the evaluation metrics. Note that in this case MRR is equivalent to MAP and P@1 is equivalent to R-Precision. For Ubuntu chat log data, since there are 10 candidate questions for each context, we additionally report P@5 and Recall@5. We study the effectiveness of the following methods:

Table 4: Experimental results for question retrieval with the Quora dataset. The best performance is highlighted in boldface. ‡ means significant difference over all the baseline methods with $p < 0.05$ measured by the Student's paired t-test.

Method	MRR	P@1
WordCount	0.786	0.659
WordCountIDF	0.811	0.699
VSM	0.833	0.737
BM25	0.861	0.781
QL	0.859	0.777
TRLM	0.865	0.778
AvgWordEmbed	0.791	0.669
CNN-Match	0.864	0.774
LSTM-CNN-Match	0.880	0.797
Combined Model	**0.894**‡	**0.819**‡

WordCount: This method computes the word co-occurrence count between the two sequences.

WordCountIDF: This method computes the word co-occurrence count weighted by IDF value between the two sequences.

VSM: This method computes the cosine similarity between the TF-IDF representation of the given two sequences.

BM25: This method computes the BM25 score between the two sequences, where we treat one of the sequences as the query and the other one as the document.

QL: This method computes the query likelihood [20] score with Dirichlet prior smoothing between the language models of the two sequences.

TRLM: This method is the translation-based language model employed by Jeon et al. [9] and Xue et al. [31]. This method has been consistently reported as the state-of-the-art method for the question retrieval task.[37].

AvgWordEmbed: This method uses the average vector of word embeddings as the sequence representation; then the cosine similarity of sequence representations is used for the candidate question ranking.

CNN-Match: This is a degenerate version of the LSTM-CNN-Match model where we remove the LSTM layer in the model, which is similar to the CDNN model proposed by Severyn and Moschitti [21].

LSTM-CNN-Match: The model presented in Section 2, which has been recently applied to other tasks, such as answer sentence selection [6, 26, 29, 32, 39].

Combined Model: We tried to combine scores of all baseline methods with neural matching models and trained a LambdaMART ranker for question ranking. This is to study whether combining learned features from basic retrieval models with neural models could lead to better retrieval performance.

3.3 Experimental Results on Question Retrieval

Table 4 shows the experimental results for the question retrieval task with the Quora dataset. We summarize our observations as

Table 5: Retrieval results for the query question "What are some good anime movies?" of different methods. The correct similar candidate question is highlighted in bold font.

Top Retrieval Results by BM25	Rank
What are good scary movies ?	1
What are some of the best anime shows?	4
Top Retrieval Results by TRLM	
What are good scary movies ?	1
What are some of the best anime shows?	2
Top Retrieval Results by LSTM-CNN-Match	
What are some of the best anime shows?	1
What is your favorite anime ?	2

follows: (1) LSTM-CNN-Match model outperforms all the baseline methods including basic retrieval models, translation model based methods and basic neural model/word embedding based methods. This shows the advantage of jointly modeling semantic match information through a neural matching model and basic word overlap information for the question retrieval task. (2) Comparing the performance of LSTM-CNN-Match model and CNN-Match model, we found that the retrieval performance will decrease if we remove the LSTM layer. This shows that modeling long term dependency in questions through LSTM is useful for boosting question search performance. (3) If we combine the learned matching score of neural models with the basic retrieval model scores, we can observe further gain over the baselines. Thus in practice the learning to rank framework is still useful for combining different features including both traditional IR model scores and the more recent neural model scores for a strong ranker for question search.

To get a better understanding of the effectiveness of the model, we checked the retrieved questions of each method. Jointly modeling term matching information with semantic matching information is important for the question retrieval task. Table 5 reports the retrieval results of different methods for the query question "What are some good anime movies?". BM25 relying on term matching between question pairs ranked the correct similar candidate question "What are some of the best anime shows?" in a relatively low position and ranked "What are good scary movies?" in the first position. TRLM suffers from a similar problem. The neural matching model LSTM-CNN-Match ranked the correct similar question candidate in the first position, since it can capture the semantic similarity between "movies" and "shows" as well as "good" and "best", which are missed by the term matching based retrieval models.

3.4 Experimental Results on Predicting Questions in Conversations

Table 6 shows the experimental results for predicting questions in conversations with the Ubuntu chat log dataset. For this task, the "Combined Model" performed the best for MRR and P@1. CNN-Match achieved the best performances for P@5 and Recall@5. We also found LSTM-CNN-Match performed worse than CNN-Match for this task. Overall neural matching models could improve the ranking effectiveness of finding questions given previous context over traditional retrieval models. Combining scores from neural matching models and traditional retrieval models could also be helpful. Our research represents an initial effort to understand the effectiveness of neural matching models for predicting questions in conversations. We find that this is a more challenging task comparing with similar question finding due to at least two reasons: 1) Unlike similar question pairs with close sequence lengths, a context is usually much longer than a candidate question in conversations. 2) The matching pattern between conversational context and candidate questions could be more complex, which is beyond semantic match or paraphrase as in question retrieval. To find more effective clues from context, more advanced model architectures like attention modeling in context should be considered. Sequence to sequence learning with an RNN Encoder-Decoder architecture [3, 22, 25] and memory networks [24] could be promising directions to explore.

Table 6: Experimental results for predicting questions in conversations with the Ubuntu chat log dataset. The best performance is highlighted in boldface. R@5 denotes Recall@5.

Method	MRR	P@1	P@5	R@5
WordCount	0.474	0.284	0.143	0.717
WordCountIDF	0.548	0.391	0.146	0.732
VSM	0.570	0.432	0.146	0.729
BM25	0.559	0.413	0.146	0.728
QL	0.483	0.337	0.127	0.633
CNN-Match	0.579	0.428	**0.155**	**0.775**
LSTM-CNN-Match	0.571	0.426	0.151	0.754
Combined Model	**0.581**	**0.440**	0.152	0.762

4 RELATED WORK

4.1 Question Retrieval

The current research for question retrieval can be divided into two categories. The first group leveraged translation models to bridge the lexical gaps between questions. Jeon et al. [9] proposed a method learning word translation probabilities from question-question pairs collected based on similar answers in CQA. Xue et al. [31] proposed a retrieval model that combines a translation-based language model for the question part with a query likelihood approach for the answer part. The translation-based language model (TRLM) has been consistently reported as the state-of-the-art method for question retrieval [37]. Topic models have also been adopted for question retrieval [34]. Recent years there are few research works on the research of building deep learning models with word embeddings for question retrieval [28, 38]. Wang et al.[28] proposed a unified framework to simultaneously handle the three problems in question retrieval including lexical gap, polysemy and word order A high level feature embedded convolutional semantic model is proposed to learn the question embeddings.

The second research group has focused on improving question search with category information about questions. Cao et al. [2] proposed a language model with leaf category smoothing for questions in the same category. Zhou et al. [37] proposed an efficient and effective retrieval model for question retrieval by leveraging user chosen categories. They achieved this by filtering some irrelevant historical questions under a range of leaf categories. Although considering category information can improve question retrieval performance, these methods could not be applied to the scenarios where the category information is not available. In many question answering and chatbot/dialogue systems, new questions issued by users have no explicit predefined category. Our work is closer to a general setting of question search where no category information are available.

4.2 Neural Conversation Models

Recent years there are growing interests on research about conversation response generation and ranking with deep learning and reinforcement learning [1, 13, 14, 22, 23, 32]. Shang et al. [22] proposed Neural Responding Machine (NRM), which is a RNN encoder-decoder framework for short text conversation and showed that it outperformed retrieved-based methods and SMT-based methods for single round conversation. Sordoni et al. [23] proposed a neural network architecture for response generation that is both context-sensitive and data-driven utilizing the Recurrent Neural Network Language Model architecture. Yan et al. [32] proposed a retrieval-based conversation system with the deep learning-to-respond schema through a deep neural network framework driven by web data. Li et al. [14] apply deep reinforcement learning to model future reward in chatbot dialogs towards building a neural conversational model based on the long-term success of dialogs. Bordes et al. [1] proposed a testbed to break down the strengths and shortcomings of end-to-end dialog systems in goal-oriented applications. They showed that an end-to-end dialog system based on Memory Networks can reach promising performance and learn to perform non-trivial operations. We work is relevant to neural conversational models. But we have different focuses on finding questions given previous conversational context.

4.3 Neural Ranking Models

A number of neural approaches have been proposed for ranking documents in response to a given query. These approaches can be generally divided into two groups: representation-focused and interaction-focused models [5]. Representation-focused models independently learn a representation for each query and candidate document and then calculate the similarity between the two estimated representations via a similarity function. As an example, DSSM [8] is a feed forward neural network with a word hashing phase as the first layer to predict the click probability given a query string and a document title.

On the other hand, the interaction-focused models are designed based on the interactions between the query and the candidate document. For instance, DeepMatch [16] is an interaction-focused model that maps each input to a sequence of terms and trains a feed-forward network to compute the matching score. These models have an opportunity to capture the interactions between query and document, while representation-focused models look at the inputs in isolation. Recently, Mitra et al. [17] proposed to simultaneously learn local and distributional representations to capture both exact term matching and semantic term matching.

All the aforementioned models are trained based on either explicit relevance judgments or clickthrough data. More recently, Dehghani et al. [4] proposed to train neural ranking models when no supervision signal is available. They used an existing retrieval model, e.g., BM25 or query likelihood, to generate large amount of training data automatically and proposed to use these generated data to train neural ranking models with weak supervision.

5 CONCLUSIONS AND FUTURE WORK

In this paper, we studied the effectiveness of neural matching models for two tasks: retrieving similar questions and predicting questions in conversations. We showed that neural matching models significantly outperforms all the baseline methods for the question retrieval task. Furthermore, when the neural matching model is combined with the basic term matching based retrieval models, we can achieve larger gains. For predicting questions in conversations, we observed that LSTM layers cannot handle long question history (past questions) and thus a simpler neural matching model with no LSTM layer outperforms all the other methods. This is a preliminary study in this area and there are still spaces to develop more advanced neural models to further improve the performance of matching conversational context with questions. For future work, we plan to continue the research on neural conversational models as a modern way for people to access information. Modeling context attentions and incorporating external knowledge into neural conversation models for finding better candidate questions could be also considered as interesting future directions.

6 ACKNOWLEDGMENTS

This work was supported in part by the Center for Intelligent Information Retrieval, in part by NSF IIS-1160894, and in part by NSF grant #IIS-1419693. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

REFERENCES

[1] Antoine Bordes and Jason Weston. 2016. Learning End-to-End Goal-Oriented Dialog. *CoRR* abs/1605.07683 (2016).
[2] Xin Cao, Gao Cong, Bin Cui, Christian Søndergaard Jensen, and Ce Zhang. 2009. The Use of Categorization Information in Language Models for Question Retrieval. In *CIKM '09*.
[3] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *CoRR* abs/1406.1078 (2014).
[4] Mostafa Dehghani, Hamed Zamani, Aliakei Severyn, Jaap Kamps, and W. Bruce Croft. 2017. Neural Ranking Models with Weak Supervision. In *SIGIR '17*.
[5] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A Deep Relevance Matching Model for Ad-hoc Retrieval. In *CIKM '16*. 55–64.
[6] Hua He and Jimmy J. Lin. 2016. Pairwise Word Interaction Modeling with Deep Neural Networks for Semantic Similarity Measurement. In *NAACL '16*.
[7] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997).
[8] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning Deep Structured Semantic Models for Web Search Using Clickthrough Data. In *CIKM '13*. 2333–2338.
[9] Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. 2005. Finding Similar Questions in Large Question and Answer Archives. In *CIKM '05*.
[10] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A Convolutional Neural Network for Modelling Sentences. *CoRR* abs/1404.2188 (2014).
[11] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[12] Oliver Lemon, Kallirroi Georgila, James Henderson, and Matthew Stuttle. 2006. An ISU Dialogue System Exhibiting Reinforcement Learning of Dialogue Policies: Generic Slot-filling in the TALK In-car System. In *EACL '06*.

[13] Jiwei Li, Michel Galley, Chris Brockett, Georgios P. Spithourakis, Jianfeng Gao, and William B. Dolan. 2016. A Persona-Based Neural Conversation Model. In *ACL'16*.

[14] Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. 2016. Deep Reinforcement Learning for Dialogue Generation. In *EMNLP'16*.

[15] Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. *CoRR* abs/1506.08909 (2015).

[16] Zhengdong Lu and Hang Li. 2013. A Deep Architecture for Matching Short Texts. In *NIPS '13*. 1367–1375.

[17] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. 2017. Learning to Match Using Local and Distributed Representations of Text for Web Search. In *WWW '17*. 1291–1299.

[18] Mikio Nakano, Noboru Miyazaki, Norihito Yasuda, Akira Sugiyama, Jun-ichi Hirasawa, Kohji Dohsaka, and Kiyoaki Aikawa. 2000. WIT: A Toolkit for Building Robust and Real-time Spoken Dialogue Systems. In *SIGDIAL '00*.

[19] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global Vectors for Word Representation. In *EMNLP '14*.

[20] Jay M. Ponte and W. Bruce Croft. 1998. A Language Modeling Approach to Information Retrieval. In *SIGIR '98*.

[21] Aliaksei Severyn and Alessandro Moschitti. Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks. In *SIGIR '15*.

[22] Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural Responding Machine for Short-Text Conversation. *CoRR* abs/1503.02364 (2015).

[23] Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A Neural Network Approach to Context-Sensitive Generation of Conversational Responses. *CoRR* abs/1506.06714 (2015).

[24] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end Memory Networks. In *NIPS'15*.

[25] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *NIPS'14*.

[26] Ming Tan, Cícero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. Improved Representation Learning for Question Answer Matching. In *ACL'16*.

[27] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A Deep Architecture for Semantic Matching with Multiple Positional Sentence Representations. In *AAAI'16*. 2835–2841.

[28] Pengwei Wang, Yong Zhang, Lei Ji, Jun Yan, and Lianwen Jin. Concept Embedded Convolutional Semantic Model for Question Retrieval. In *WSDM '17*.

[29] Shuohang Wang and Jing Jiang. 2016. A Compare-Aggregate Model for Matching Text Sequences. *CoRR* abs/1611.01747 (2016).

[30] Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral Multi-Perspective Matching for Natural Language Sentences. *CoRR* abs/1702.03814 (2017). http://arxiv.org/abs/1702.03814

[31] Xiaobing Xue, Jiwoon Jeon, and W. Bruce Croft. 2008. Retrieval Models for Question and Answer Archives. In *SIGIR '08*.

[32] Rui Yan, Yiping Song, and Hua Wu. 2016. Learning to Respond with Deep Neural Networks for Retrieval-Based Human-Computer Conversation System. In *SIGIR'16*. 55–64.

[33] Liu Yang, Qingyao Ai, Jiafeng Guo, and W. Bruce Croft. 2016. aNMM: Ranking Short Answer Texts with Attention-Based Neural Matching Model. In *CIKM '16*. 287–296.

[34] Liu Yang, Minghui Qiu, Swapna Gottipati, Feida Zhu, Jing Jiang, Huiping Sun, and Zhong Chen. 2013. CQArank: Jointly Model Topics and Expertise in Community Question Answering. In *CIKM '13*.

[35] Steve J. Young, Milica Gasic, Blaise Thomson, and Jason D. Williams. 2013. POMDP-Based Statistical Spoken Dialog Systems: A Review. *Proc. IEEE* 101, 5 (2013), 1160–1179.

[36] Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. 2014. Deep Learning for Answer Sentence Selection. *CoRR* abs/1412.1632 (2014).

[37] Guangyou Zhou, Yubo Chen, Daojian Zeng, and Jun Zhao. 2013. Towards Faster and Better Retrieval Models for Question Search. In *CIKM '13*.

[38] Guangyou Zhou, Tingting He, Jun Zhao, and Po Hu. Learning Continuous Word Embedding with Metadata for Question Retrieval in Community Question Answering. In *ACL'15*.

[39] Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. 2016. Text Classification Improved by Integrating Bidirectional LSTM with Two-dimensional Max Pooling. *CoRR* abs/1611.06639 (2016).