A NeuRetrieval Model for Human-Computer Conversations

Rui Yan
Institute of Computer Science
and Technology (ICST)
Peking University
Beijing 100871, China
ruiyan@pku.edu.cn

Dongyan Zhao
Institute of Computer Science
and Technology (ICST)
Peking University
Beijing 100871, China
zhaody@pku.edu.cn

ABSTRACT

To establish an automatic conversation system between human and computer is regarded as one of the most hardcore problems in computer science. It requires interdisciplinary techniques of information retrieval, natural language processing, data management as well as artificial intelligence. The arrival of big data era reveals the feasibility to create a conversation system empowered by data-driven approaches. Now we are able to collect extremely large conversational data on Web, and organize them to launch a human-computer conversation system. Owing to the diversity of Web resources available, a retrieval-based conversation system will be able to find at least some responses from the massive data repository for any user inputs. Given a human issued utterance, i.e., a query, a retrieval-based conversation system will search for appropriate replies, conduct a relevance ranking, and then output the highly relevant one as the response. In this paper, we propose a novel retrieval model named NeuRetrieval for short text understanding, representation and semantic matching. The proposed model is general and unified for both single-turn and multi-turn conversation scenarios in open domain. In the experiments, we investigate the effectiveness of the proposed deep neural network model for human-computer conversations. We demonstrate performance improvement against a series of baseline methods in several evaluation metrics. In contrast with previously proposed methods, NeuRetrieval is tailored for conversation scenarios and demonstrated to be more effective.

KEYWORDS

Conversation system; neural networks; retrieval model

ACM Reference Format:
Rui Yan and Dongyan Zhao. 2018. A NeuRetrieval Model for Human-Computer Conversations. In *Proceedings of The 2018 Web Conference Companion (WWW'18 Companion).* ACM, New York, NY, USA, 8 pages. https://10.1145/3184558.3186341

WWW'18 Companion, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5640-4/18/04.
https://10.1145/3184558.3186341

1 INTRODUCTION

To have an intelligent virtual assistant and/or chat companion through a conversation system using natural languages has always been a long cherished goal for researchers and practitioners. A significant amount of efforts have been devoted to this area for decades, and promising achievements have been gradually achieved so that we can expect real conversation applications in real life, rather than in Sci-Fi movies or research labs only. The demand for conversation systems leads to cutting-edge technology in the spotlight from both academia and industry. Creating a smart human-computer conversation system is no longer a fantasy.

It is believed to be challenging for computers to maintain a relevant, meaningful and continuous conversation with humans. To have conversations with humans generally involves interdisciplinary techniques such as information retrieval, natural language processing, data management, as well as artificial intelligence. The arrival of big data era also accelerates the development of human-computer conversation studies. Owing to tons of public resources for conversations available on the Web, we shall learn what to respond given (almost) any inputs by retrieving materials from a conversational data repository. It is time to build data-driven conversation systems between human and computer.

Building conversation systems, in fact, has attracted many attentions over the past decades. In early years, researchers have investigated into task-oriented conversation systems [3, 24, 44], which are basically for vertical domains. The conversational inputs are restricted and predictable; hence it would be easier—compared with open-domain systems—to design the logic, create the rules, prepare the data and construct the candidate reply to handle the particular task [42]. For instance, in a conversation system for flight booking or bus route inquiring, the computer side only needs to capture the origin, destination and flight/bus information, and then respond accordingly with templates [44]. One of the most obvious limitations of task-specific service is that the conversation cannot exceed the system topic scope. Illegible inputs will not be accepted, which is regarded as a hard constraint. The underlying system design philosophy is nearly impossible to generalize to the open domain.

It is only recently that researchers focus on non-task-oriented (i.e., open-domain) conversation systems for their functional, social, and entertainment roles in real-world applications [4, 5, 10, 12, 16, 18, 21, 36, 39, 41]. Creating an open-domain conversation system to interact with humans

is an interesting but notoriously challenging problem. Since people are free to say anything to the system, it is impossible to prepare the interaction logic and domain knowledge, which can be, in contrast, specified in task-specific systems before hand. Besides, the number of possible combinations of conversation status are literally infinite, so that conventional hand-crafted rules and templates would fail [31, 37].

Along with the maturity of Web 2.0, there has been an explosion in the number of people having public conversations on websites such as Bulletin Board System (BBS) forums, social media (e.g., Facebook, Twitter) and community question answering (cQA) platforms (e.g., Baidu Zhidao, Yahoo! Answers). These resources provide a unique opportunity to build collections of naturally occurring conversations that are orders of magnitude larger than those previously available. They also propel the development of retrieval-based techniques in the field of open-domain conversation research. The merit is that, owing to the diversity on the Web, the system will be able to retrieve at least some responses for any user input.

The big data era, however, seems like a double-edged sword. On one side, it brings the great opportunity, as mentioned above, to build practical human-computer conversation systems in open domain. On the other side, there are also challenges. Given a user-issued query, we need to find appropriate candidate replies from a very large volume of data. In this paper, we propose a new retrieval model based on deep neural network structures, namely "NeuRetrieval". The intuition is mani-fold. Firstly, neural networks are demonstrated to be effective to represent short texts. For short text conversations, abstractive representation by neural networks could be a natural solution for retrieval tasks. Secondly, there are two typical scenarios for human-computer conversation: single-turn and multi-turn. For multi-turn conversations, there is always additional information to be used such as "contexts" (a.k.a. previous utterance sentences in a continuous conversation session). The proposed method should be general for both single-turn and multi-turn conversation scenarios, while capture and integrate as much information as possible. It is less feasible and less practical to apply keyword-based search for an entire conversation session with multiple utterances. In this case, neural networks are more powerful to characterize a condensed representation especially for multi-turns. We use neural networks to formulate the NeuRetrieval model to represent, to match and hence to retrieve replies accordingly. The proposed model is tailored for human-computer conversation. A deep neural network (DNN)-based retrieval model tells how likely each reply is appropriate to respond the query given the context (if any). We retrieve highly relevant replies.

We conduct extensive experiments for conversation setups between humans and computers. In particular, we build the system upon an extremely large conversation resource, i.e., almost 10 million pairs of atomic human conversations. We then run experiments against several other rival algorithms to verify the effectiveness of the neural retrieval model. Our system (generally) outperforms traditional and recent baselines regarding a variety of different evaluation metrics in terms of p@1, MAP, nDCG and MRR. The result indicates that our conversation system provides a novel and useful insight to facilitate human-computer conversations.

To sum up, our contributions are novel in following ways:

We propose a new retrieval model named "NeuRetrieval" for the human-computer conversation system. The NeuRetrieval model is demonstrated to be effective to represent, to match, and hence to retrieve short texts for conversations with deep neural networks.

There are two typical scenarios for human-computer conversations: 1) single-turn and 2) multi-turn. The proposed model is general and well unified to adapt for both scenarios. Different structures in conversation sessions are also investigated in this study.

The rest of the paper is organized as follows. We start by reviewing related work. In Sections 3 and 4, we describe the framework of conversation system and task formulation. We introduce the detailed mechanisms of the NeuRetrieval model in single-turn and multi-turn conversations. We devise experimental setups and evaluations against a variety of baselines and discuss results in Section 5. Finally we draw conclusions in Section 6.

2 RELATED WORK

2.1 Conversation Systems

Early work on conversation systems is generally based on rules or templates and is designed for specific domains [3, 24]. These rule-based approaches require no data or few data for training, while instead require many manual efforts to build the model, or to handcraft rules, which is usually very costly. The conversation structure and status tracking in vertical domains are more feasible to learn and to infer [44]. However, the coverage of such systems are also far from satisfaction. Later, people begin to pay more attention to automatic conversation systems in open domains [18, 21].

From specific domains to the open domain, the need for a big amount of data is increasing substantially to build a conversation system. As information retrieval techniques are developing fast, researchers obtain promising achievements in (deep) question and answering systems. In this way, an alternative approach is to build a conversation system with a knowledge base consisting of a number of question-answer pairs. Leuski *et al.* [1] build systems to select the most suitable response to the current message from the question-answer pairs using a statistical language model in cross-lingual information retrieval, but have a major bottleneck of the creation of the knowledge base (i.e., question-answer pairs) [19]. Researchers propose to augment the knowledge base with question-answer pairs derived from plain texts [7, 23]. The number of resource pairs can be, to some extent, expanded, but are still relatively small.

Nowadays, with the prosperity of social media and other Web 2.0 resources, such as community question and answering (cQA) or microblogging services, a very large amount of conversation data become available [16]. A series of

information retrieval-based methods are applied to short text conversation using microblog data [6, 10, 36]. Higashinaka *et al.* also combine template generation with the search-based methods [18]. Ritter *et al.* have investigated the feasibility of conducting short text conversation by using statistical machine translation (SMT) techniques, as well as millions of naturally occurring conversation data in Twitter [12]. In the approach, a response is generated from a model, not retrieved from a repository, and thus it cannot be guaranteed to be a legitimate natural language text.

In previous studies of conversation systems, few approaches take into account the structure information in the conversation stream, especially in multi-turn conversations. Now we provide a general and unified solution based on the deep neural network architecture tackling both single-turn and multi-turn conversations. Besides, structure information in the conversation flow is also incorporated. The approach shows a novel insight in conversation systems.

2.2 Deep Neural Networks

In recent years, deep neural networks (DNNs, also known as *deep learning*) have made significant improvement in Natural Language Processing. DNNs are highly automated learning machines; they can extract underlying abstract features of data automatically by exploring multiple layers of non-linear transformation [2].

In NLP models, a word typically acts as an atomic unit. However, words are discrete by nature; it seems nonsensical to feed word indexes to DNNs. A typical approach is to map a discrete word to a dense, low-dimensional, real-valued vector, called an *embedding* [22]. Each dimension in the vector captures some aspect of underlying word meanings.

Prevailing DNNs for sentence-level modeling include convolutional neural networks (CNNs) and recurrent neural networks (RNNs). In CNNs, we have a fixed-size sliding window to capture local patterns of successive words [14], whereas RNNs keep one or a few hidden states, and collect information along the word sequence in an iterative fashion [35]. The multi-layer RNN structure has been investigated for document-paragraph modeling [20]. Socher *et al.* leverage sentence parse trees and build recursive networks [30]. However, conversational utterances are usually casual, and hence recursive models are less applicable.

Beyond a single sentence, some studies are aimed to capture the relationship between two sentences—known as sentence pair modeling—with applications like paraphrase detection [9], discourse unit recognition [43], textual entailment recognition [26], etc. A sentence-pair DNN model is typically built upon underlying sentence-level models (CNNs/RNNs). Then two sentences' information is combined by matching heuristics like concatenation, cosine measure, or inner-product [9]. Hu *et al.* develop word-by-word matching approaches [11], and obtain a similarity matrix between two sentences. Recently, Rocktäschel *et al.* propose context-aware matching approaches [26], where the first sentence's information is available when modeling the second one. Such context-awareness interweaves individual sentence modeling and sentence matching, prohibiting pre-calculating the vector representation of a sentence; hence these methods are considerably more computational intensive. For efficiency consideration, we choose to leverage vector concatenation, which is simple yet effective.

We revisit recent context-aware method from both retrieval-based systems or generative conversation systems. They are all based on sentence sequences with orders [27, 38] or without orders [32, 37]. Contextual utterances are modeled with hierarchies on word- and sentence-levels [27, 34, 45] or without hierarchies at all [32, 37, 38]. The proposed NeuRetrieval model is more than traditional matching and ranking. The conversation is streamed as utterance stream flows, and there are contextual information needs to be incorporated in a multi-turn conversation scenario. The proposed deep retrieval model utilizes context information tailored for human-computer conversation, and we make thorough comparisons in the experimental evaluations.

3 METHODOLOGY

3.1 Problem Formulation

The research problem of automatic human-computer conversation is defined as one or more turns of interactions between human and computer. Within each turn of the conversation, given the message issued from the human, the computer would provide a reply in response to the coming message. To this end, given the user message as a query q, our system searches for similar postings p from the vast repository of conversational data. The associated replies r of the obtained postings p within a $\langle$*posting-reply*$\rangle$ pair is fed into the representation and matching part. Finally, the best matched reply r will be returned as an appropriate response to output. As we propose to suit both scenarios of single-turn and multi-turn conversations, we manage to use the context of previous utterances $\mathcal{C}=\{s_1, s_2, \dots\}$ in the conversation session as well.

We would firstly introduce query representation, which is actually an encoder, using query with or without context information. After we encode the query as a representation vector, we also learn to encode the candidate replies. Next we match the query and reply through the deep neural networks.

3.2 Query Representation

In recent years, deep neural networks (DNNs, also known as *deep learning*) have made significant improvement. With big data available, DNNs are highly automated learning machines; they can extract underlying abstract features of data automatically by exploring multiple layers of non-linear transformation [2]. We first give a quick overview of word embedding and the neural network structure in Long-Short Term Memory (LSTM).

Word Embeddings. Traditional models usually treat a word as a discrete token; thus, the internal relation between similar words would be lost. Word embedding [22] is a standard apparatus in neural network-based text

processing. A word is mapped to a low dimensional, real-valued vector. This process, known as vectorization, captures some underlying meanings. Given enough data, usage, and context, word embeddings can make highly accurate guesses about the meaning of a particular word. Embeddings can equivalently be viewed that a word is first represented as a one-hot vector and multiplied by a look-up table [22].

In our model, we first vectorize all words using their embeddings, which serve as the foundation of our deep neural networks. Word embeddings are initialized randomly, and then tuned during training as part of model parameters.

Bi-Directional LSTM. We use a bi-directional long short term memory (Bi-LSTM) recurrent network to propagate information along the word sequence. A recurrent neural network (RNN) keeps a hidden state vector, which changes according to the input in each time step. As RNNs can iteratively aggregate information along a sequence, they are naturally suitable for sentence modeling.

LSTM is an advanced type of RNN by using memory cells and gates to learn long term dependencies within a sequence [25, 33]. LSTM models are defined as follows: given a sequence of inputs, an LSTM associates each position with *input*, *forget*, and *output gates*, denoted as i_t, f_t, and o_t respectively. The vector l_t is used to additively modify the memory contents. Given an input sentence $s = \{x_0, x_1, \ldots, x_T\}$, where x_t is a sequence of inputs while e_t denotes the vector for embedding of individual unit (i.e., word or sentence) at position t in the sentence. LSTM outputs a representation h_t for position t by combining h_{t-1} and e_t, given by

$$\begin{aligned} \begin{bmatrix} i_t \\ f_t \\ o_t \\ l_t \end{bmatrix} &= \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{bmatrix} W \cdot \begin{bmatrix} h_{t-1} \\ e_t \end{bmatrix} \\ \tilde{h}_t &= f_t \cdot \tilde{h}_{t-1} + i_t \cdot l_t \\ h_t^s &= o_t \cdot \tilde{h}_t \end{aligned} \tag{1}$$

where $\tilde{h}$ is an auxiliary variable and can be viewed as the information stored in memory cell. $\sigma(\cdot) = \frac{1}{1+e^{-\cdot}}$ is a known as a sigmoid/logistic function.

A single directional LSTM typically propagates information from the first word to the last; hence the hidden state at a certain step is dependent on its previous words only and is thus blind of future words. The variant Bi-LSTM [8] is proposed to utilize both previous and future words by two separate RNNs, propagating forward and backward, and generating two independent hidden state vectors $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$, respectively. The two state vectors are concatenated to represent the meaning of the t-th word in the sentence, i.e., $h_t = \left[\overrightarrow{h_t}; \overleftarrow{h_t}\right]$.

3.2.1 Single-Turn Query Encoder. Here we denote a query as a sequence of tokens, i.e., words. Each word is associated with its d-dimensional embedding. We need to represent the query as a d-dimensional representation. An encoder is a neural model where output units are directly connected with or identical to input units. Typically, inputs are compressed into a representation using neural models (encoding). For a single-turn query encoder, the input is the query only.

For simplicity, we define Bi-LSTM(h_{t-1}, e_t) to be the Bi-LSTM operation on vectors h_{t-1} and e_t to achieve h_t.

For the single-turn representation, the query is treated as one sequence of tokens. Following Sutskever *et al.* [33], we trained an encoder that first maps input documents into vector representations from a Bi-LSTM encoder. No context information will be considered (Model 1, Figure 1).

3.2.2 Multi-Turn Query Encoder. We now take the context information into account in order to formulate an enriched representation of queries with contexts, a.k.a., multi-turn representations. An intuitive solution is a simple extension of the single-turn representation: we incorporate multi-utterance information (i.e., different sentences). Note that each sentence ends up with a special end-of-sentence symbol $\langle eos \rangle$. For clarification, we use notations as:

- h_t^w and h_t^s denote hidden vectors from Bi-LSTM models, the subscripts of which indicate time step t, the superscripts of which indicate operations at word level (w) or utterance sentence level (s). h_t^s (enc) specifies encoding stage.
- e_t^w and e_t^s denotes word-level and sentence-level embedding for word and sentence at position t in terms of its residing sentence or session context.

The simple extension of multi-turn representation treats all the input as a long sequence of tokens: all the utterances as well as the query is concatenated from head to tail. Hence we can also train an encoder using the Bi-LSTM operation. Here sentence structures are utilized as a plain model, as shown in Model 2, Figure 1. The calculation is defined as:

$$h_t^s(enc) = \text{Bi-LSTM}_{encode}^{sentence}(e_t^w, h_{t-1}^w(enc)) \tag{2}$$

We notice that words creates a joint meaning of a sentence, the juxtaposition of sentences also creates a joint meaning of a conversation session. Such an observation indicates a hierarchical structure between words and sentences. We hence propose a hierarchical multi-turn representation for queries and the associated context information. We first obtain representation vectors at the sentence level by putting one layer of LSTM (i.e., $\text{LSTM}_{encode}^{word}$) on top:

$$h_t^w(enc) = \text{Bi-LSTM}_{encode}^{word}(e_t^w, h_{t-1}^w(enc)) \tag{3}$$

The vector output at the ending time-step is used to represent the entire utterance sentence. To build representation for the current conversation session with multiple turns, another layer of LSTM (denoted as $\text{LSTM}_{encode}^{sentence}$) is placed on top of all sentences, computing representations sequentially:

$$h_t^s(enc) = \text{Bi-LSTM}_{encode}^{sentence}(e_t^s, h_{t-1}^s(enc)) \tag{4}$$

Thus one Bi-LSTM operates at the word level, leading to the acquisition of sentence-level representations that are then used as inputs into the second Bi-LSTM that acquires higher-level representations in a hierarchical structure (shown in Model 3, Figure 1).

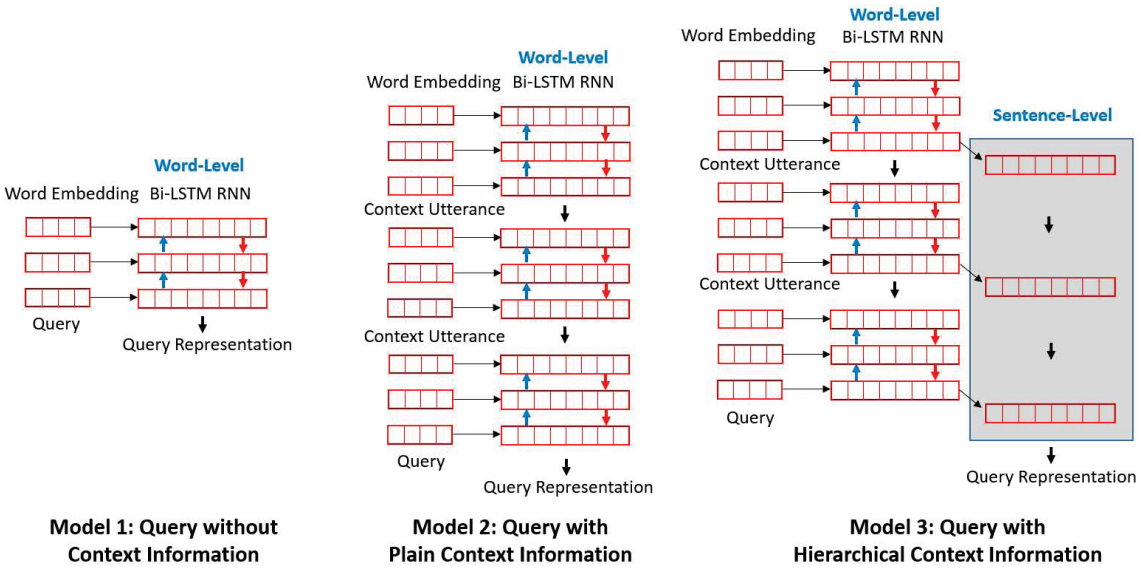

Figure 1: Different models with and without context information. For Model 1 and Model 2, we have word-level representations while for Model 3, we have both word-level and sentence-level representations.

3.3 Matching

As mentioned, we need to compare the candidate replies with the query (and the context). The scoring function outputs a scalar in $\mathbb{R}$ (appropriateness or inappropriateness) for a particular candidate, given either the query itself or with contexts. The matching scores are computed by the deep neural network architecture.

Our sentence models based on Bi-LSTM learn to encode the replies and queries (with or without contexts) as vectors, which can then be used to compute their similarity. These are then used to compute a *query-reply* similarity score, which together with the query and reply vectors are joined in a single representation.

On the basis of sentence representations using Bi-LSTM, we can model the interactions between queries and replies. Given the output of sentences for processing queries and replies, their resulting vector representations $\mathbf{x}_q$ and $\mathbf{x}_r$, can be used to compute a *query-reply* similarity score. We follow the approach of [28] that defines the similarity between $\mathbf{x}_q$ and $\mathbf{x}_r$ vectors as follows:

$$sim(\mathbf{x}_q, \mathbf{x}_r) = \mathbf{x}_q^T M \mathbf{x}_r \tag{5}$$

where $M \in \mathbb{R}^{d \times d}$ is a similarity matrix. Here, we seek a transformation of the candidate posting $\mathbf{x}'_r = M\mathbf{x}_r$ that is the closest to the input query $\mathbf{x}_q$. The similarity matrix M is a parameter of the network and is optimized during training.

After the match using the similarity matrix M, Equation (5) produces a single score x_{sim} capturing various aspects of similarity (syntactic and semantic) between the input information. The joint layer concatenates all intermediate vectors as well as the similarity score into a single vector:

$$\mathbf{x}_{join} = [\mathbf{x}_q^T; \mathbf{x}_{sim}; \mathbf{x}_r^T]$$

Then the concatenated vectors are fed to an ensuing network for further information mixing. Vector concatenation for sentence matching is also applied in other studies like [43], which is effective yet of low complexity order, compared with other word-by-word matching [11].

The joint vector is then passed through a 3-layer, fully-connected, feed-forward neural network, also known as *multi-layer perceptron* (MLP) [2], which allows rich interactions between a vector pair. The network enables to extract features automatically, starting from lower-level representations to higher-level ones. Finally, a single neuron outputs the score between a query (or the context) and a reply. The final scoring neuron is essentially a linear regression.

Ranking problems can apply pairwise ranking loss such as hinge loss or cross-entropy loss. Here we apply hinge loss to train our DNN network. Given a triple score(q, r^+) in the training set, we randomly sample a negative instance r^-. The objective is to maximize the scores of positive samples while minimizing that of the negative samples. Concretely, we would like score(q, r^+) to be as least score(q, r^-) plus a margin Δ. The training objective is to

$$\min_{\Omega} \sum_{q, r^+} \max\left\{0, \Delta + \mathcal{F}(q, r^+) - \mathcal{F}(q, r^-)\right\} + \lambda \|\Omega\|_2^2 \tag{6}$$

Neural networks have a large capacity to learn complex decision functions they tend to easily over-fit. To mitigate the over-fitting issue we augment the cost function with an ℓ_2 penalty with coefficient λ for all the parameters Ω which are weight and bias values optimized by the network.

4 EXPERIMENTS AND EVALUATION

In this section, we evaluate our proposed model for the conversation task against a series of baselines given a huge conversation resource. The objectives of our experiments are 1) to evaluate the effectiveness of our proposed NeuRetrieval model compared a series of baselines, and 2) to evaluate contextual information usage for query representation.

4.1 Dataset

We use the data which contain a large number of human conversations crawled from open Web, where the users publish a posting visible to the public, and then receive a bunch of subsequent replies to their utterances. We conducted data filtering and cleaning procedures by removing extremely short utterances and those of low linguistic quality as indicated in [15, 37, 40]. In total, we have 9,023,854 virtual documents of (*posting*, *reply*) pairs extracted [37].

In the retrieval repository, a subsequent *reply* has a responding relationship to the antecedent *posting*, and each pair can be regarded as an atomic conversation of two utterances. The data repository is demonstrated to be a rich resource to facilitate human-computer conversations based on retrieval [11, 16, 37].

4.2 Experimental Setups

4.2.1 Evaluation Metrics. Since we have labeled replies for test queries from the data [37], given the ranking lists, we evaluated the performance in terms of the following metrics: precision@1 (p@1), mean average precision (MAP) [21], and the normalized discounted cumulative gain (nDCG) [13]. Since the system outputs the best selected reply, p@1 is the precision at the 1st position, and should be the most natural way to indicate the fraction of suitable responses among the top-1 reply retrieved. Besides, we provided a top-k ranking list for the test queries using nDCG and MAP, which test the potential for a system to provide more than one appropriate response as candidates. We aimed at selecting as many appropriate responses as possible into the top-k list and rewarding methods that return suitable replies.

Unlike MAP and nDCG, which examine the ranks of appropriate candidates, Mean Reciprocal Rank (MRR) focuses on evaluating the capability of retrieval systems to find (perhaps) the best result which means the positive triples created by humans in our dataset. MRR is useful but does not test the full capability because there can be multiple appropriate utterances to continue a conversation.

4.2.2 Algorithms for Comparison. To illustrate the performance of our approach, we include several alternative algorithms as baselines for comparison. The baselines can be divided into two categories, i.e., 1) generation-based methods and 2) retrieval-based methods for conversation systems from very recent studies. Since our proposed approach is technically a retrieval-based method, we mainly focus on the second category. For fairness we conducted the same preprocessing procedures and data cleaning for all algorithms.

Generation-based Conversation. For this group of algorithms, the conversation system will generate a response from a given input, i.e., a query from the user under the conversational scenario.

- *Statistical Machine Translation (SMT)*: SMT is a machine translation paradigm which "translates" a query into a "reply". We implemented the phrase-based translation for the conversation modeling in [12].
- *LSTM-RNN*: LSTM-RNN is basically a Recurrent Neural Network (RNN) using the Long Short Term Memory (LSTM) architecture. The RNN with LSTM units consists of memory cells in order to store information for extended periods of time. We use LSTM-RNN for both generation and retrieval baselines. For generation, we first use an LSTM-RNN to encode the input sequence (query) to a vector space, and then use another LSTM-RNN to decode the vector into the output sequence (reply) [33]; for retrievals, we adopt the LSTM-RNN to construct sentence representations and use cosine similarity to output the matching score [25].
- *Neural Responding Machine.* We implement the neural responding machine (NRM) proposed in [29], which is an RNN-based generation approach.

Retrieval-based Conversation. The approaches within this group of baselines are based on retrieval systems, which return the best matched candidate reply out of the conversational repository given a particular query.

- *Okapi BM25.* We include the standard retrieval technique to rank candidate replies. For each query, we find the most relevant reply using BM25 model [17] from the corpus.
- *DeepMatch.* The DeepMatch method considers multiple granularity from the perspective of topics via LDA [10].
- *ARC-CNN.* This approach is a CNN based method with convolutionary layers which construct sentence representations and produce the matching scores via a MLP layer [11].
- *Rank Optimized Conversation Framework (ROCF).* The ROCF uses context information [38], which aims at retrieving more appropriate replies based on contexts from previous turns. It is a combination of context-insensitive ranking and context-aware ranking.
- *Deep Learning-to-Respond (DL2R).* The DL2R uses a query reformulation framework [37]. The query reformulation is based on different context utilization strategies.
- *NeuRetrieval.* We propose the NeuRetrieval system for short-text conversation between human and computer. We have some novel insights by investigating 1) the sequential modeling of contextual information for query representation in multi-turn conversations, 2) a plain/hierarchial structure for utterance fusion, and 3) a deep learning framework for semantic matching given the learned representations.

4.3 Overall Performance

We compare the performance of all methods including baselines and our proposed NeuRetrieval model measured in terms of MAP and nDCG, shown in Table 1.

For the generative methods, the baselines provide one generation as the response to output. Hence we do not

Table 1: Retrieval performance against baselines. For generative methods, they generate one response given each query. Hence the p@1 in fact refers to accuracy. Other metrics are not applicable. Note that there are variants for the *NeuRetrieval* method, denoting single-turn representation, multi-turn representation with plain structure or with hierarchical structure (i.e., Model 1, Model 2 and Model 3).

Generative Methods	p@1	MAP	nDCG@5	nDCG@10	nDCG@20	MRR
SMT [12]	0.363					
LSTM-RNN [33]	0.441					
NRM [29]	0.465					
Context-Insensitive Retrieval	**p@1**	**MAP**	**nDCG@5**	**nDCG@10**	**nDCG@20**	**MRR**
Okapi BM25	0.272	0.253	0.337	0.302	0.368	0.169
DeepMatch [10]	0.457	0.317	0.419	0.454	0.508	0.275
LSTM-RNN [25]	0.338	0.283	0.330	0.371	0.431	0.228
ARC-CNN [11]	0.394	0.294	0.397	0.421	0.477	0.232
Context-Aware Retrieval	**p@1**	**MAP**	**nDCG@5**	**nDCG@10**	**nDCG@20**	**MRR**
ROCF [38]	0.711	0.412	0.651	0.666	0.702	0.321
DL2R [37]	0.731	0.416	0.663	0.682	0.717	0.333
NeuRetrieval (Model 1)	0.435	0.320	0.404	0.456	0.498	0.265
NeuRetrieval (Model 2)	0.721	0.411	0.660	0.657	0.708	0.331
NeuRetrieval (Model 3)	0.735	0.418	0.669	0.683	0.715	0.338

compare MAP or nDCG for this algorithm group. Note that the original response is not likely to be generated; thus it is infeasible to calculate the MRR. In general, the generative algorithms have relatively high p@1 scores, but the generated responses are ambiguous or broad but not specific enough. Such responses might be relevant but not appropriate enough for conversation [29, 37].

We can see great improvement for NeuRetrieval against original retrieval-based baselines. *Okapi BM25* represents the standard (and simple) retrieval system. The performance for BM25 is not as good as the other deep learning-based retrieval systems. Deep learning systems are proved to have strong capabilities to learn the abstractive representation [2, 14, 30]. BM25 only utilizes the shallow representation of term-level processing. The deep learning algorithm groups in general overwhelm shallow learning algorithms. The observation is similar for both the context insensitive retrieval group and the context-aware retrieval group.

The context-aware methods outperform the standard deep learning baselines. The benefits are believed to be due to the context information, while the other deep learning baselines are matching metrics for single turn conversations only.

We compare the context-aware retrieval methods in details. The proposed NeuRetrieval model shows a better performance against context-aware baselines. The difference is that we change to a new context modeling method: we use a sequential modeling rather than a simple integration or a combination of all possible utterances. The structure of conversation session is also incorporated in the NeuRetrieval model. Sequential and hierarchical modeling for conversation streams might result in the improvement.

We have different ways to use contextual information via contextual query representation as shown in Figure 1. *Model 1* actually degenerates to a single-turn conversation scenario. No context information is integrated. There are two different ways to incorporate context information, either *Model 2*, which does not distinguish word-level and sentence-level information, or *Model 3*, which models the word-level and sentence-level information in two hierarchies.

We can see that the improvement between *Model 1* and *Model 2* or *Model 3* is rather obvious, indicating that context information is quite beneficial to retrieve better candidates especially under the conversational scenarios. The results show that *Model 3* outperforms *Model 2*. The results indicate a proper way to characterize context information is important. The hybrid modeling of word and sentence information in different hierarchies is demonstrated to be better.

5 CONCLUSIONS

In this paper, we propose to establish a new retrieval method for search-based human-computer conversation system. Given a human-issued utterance as the query, our proposed system will return the corresponding responses based on a *NeuRetrieval* model using deep neural networks. There are three major contributions in this work: 1) we propose a contextual query encoder with sequential information captured for the conversation task; 2) we investigated different context representation strategies, with or without hierarchical structures; 3) we establish the deep neural network architecture featured with above strategies and components.

We examine the effect of our proposed NeuRetrieval model with several baselines using a series of evaluation metrics. Our method (generally) outperforms strong baselines. In general, context information is demonstrated to be useful for conversations, especially for multi-turn conversations. Hierarchical modeling for the context and the query is also helpful. In the future, we will investigate more features for further improvements in performance.

ACKNOWLEDGMENTS

We thank the reviewers for their insightful comments. This work was supported by the National Key Research and Development Program of China (No. 2017YFC0804001), the National Science Foundation of China (No. 61672058). Rui Yan was sponsored by CCF-Tencent Open Research Fund and Microsoft Collaborative Research Program.

REFERENCES

[1] Pear Analytics. 2009. Twitter Study–August 2009. 15 (2009).
[2] Yoshua Bengio. 2009. Learning deep architectures for AI. *Foundations and Trends in Machine Learning* 2, 1 (2009), 1–127.
[3] Léon Bottou. 1998. Online learning and stochastic approximations. *On-line learning in neural networks* 17 (1998), 9.
[4] Miguel A Carreira-Perpinan and Geoffrey E Hinton. 2005. On contrastive divergence learning. In *Artificial Intelligence and Statistics*, Vol. 2005. 17.
[5] George Casella and Edward I George. 1992. Explaining the Gibbs sampler. *The American Statistician* 46 (1992), 167–174.
[6] Kailong Chen, Tianqi Chen, Guoqing Zheng, Ou Jin, Enpeng Yao, and Yong Yu. 2012. Collaborative Personalized Tweet Recommendation. In *SIGIR '12*. 661–670.
[7] Gao Cong, Long Wang, Chin-Yew Lin, Young-In Song, and Yueheng Sun. 2008. Finding Question-answer Pairs from Online Forums. In *SIGIR'08*. 467–474.
[8] Alan Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *Proc. Acoustics, Speech and Signal Processing*. 6645–6649.
[9] Hua He, Kevin Gimpel, and Jimmy Lin. 2015. Multi-perspective sentence similarity modeling with convolutional neural networks. In *EMNLP*. 1576–1586.
[10] John Hopcroft, Tiancheng Lou, and Jie Tang. 2011. Who Will Follow You Back?: Reciprocal Relationship Prediction. In *CIKM '11*. 1137–1146.
[11] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *NIPS*. 2042–2050.
[12] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated Gain-based Evaluation of IR Techniques. *ACM Trans. Inf. Syst.* 20, 4 (2002), 422–446.
[13] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated Gain-based Evaluation of IR Techniques. *ACM Trans. Inf. Syst.* 20, 4 (2002), 422–446.
[14] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188* (2014).
[15] Ross Kindermann, James Laurie Snell, et al. 1980. *Markov random fields and their applications*. Vol. 1. American Mathematical Society Providence, RI.
[16] Tsung-Ting Kuo, Rui Yan, Yu-Yang Huang, Perng-Hwa Kung, and Shou-De Lin. 2013. Unsupervised Link Prediction Using Aggregative Statistics on Heterogeneous Social Networks. In *KDD '13*. 775–783.
[17] John Lafferty and Chengxiang Zhai. 2001. Document Language Models, Query Models, and Risk Minimization for Information Retrieval. In *SIGIR '01*. 111–119.
[18] Victor Lavrenko and W. Bruce Croft. 2001. Relevance Based Language Models. In *SIGIR '01*. 120–127.
[19] Anton Leuski and David Traum. 2011. NPCEditor: Creating virtual human dialogue using information retrieval techniques. *AI Magazine* 32, 2 (2011), 42–56.
[20] Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015. A Hierarchical Neural Autoencoder for Paragraphs and Documents. In *ACL-IJCNLP'15*. 1106–1115.
[21] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*. Vol. 1. Cambridge University Press.
[22] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[23] Elnaz Nouri, Ron Artstein, Anton Leuski, and David R Traum. 2011. Augmenting Conversational Characters with Generated Question-Answer Pairs.. In *AAAI Fall Symposium: Question Generation*.
[24] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank citation ranking: bringing order to the web. (1999).
[25] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2015. Deep Sentence Embedding Using the Long Short Term Memory Network: Analysis and Application to Information Retrieval. *arXiv preprint arXiv:1502.06922* (2015).
[26] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2015. Reasoning about Entailment with Neural Attention. *arXiv preprint arXiv:1509.06664* (2015).
[27] Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models. In *AAAI'16*. 3776–3783.
[28] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks. In *SIGIR '15*. 373–382.
[29] Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural Responding Machine for Short-Text Conversation. In *ACL-IJCNLP'15*. 1577–1586.
[30] Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *EMNLP'11*. 151–161.
[31] Fei Song and W. Bruce Croft. 1999. A General Language Model for Information Retrieval. In *CIKM '99*. 316–321.
[32] 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. In *NAACL'15*. 196–205.
[33] Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*. 3104–3112.
[34] Zhiliang Tian, Rui Yan, Lili Mou, Yiping Song, Yansong Feng, and Dongyan Zhao. 2017. How to Make Context More Useful? An Empirical Study on Context-Aware Neural Conversational Models. In *ACL'17*. 231–236.
[35] Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015. Classifying relations via long short term memory networks along shortest dependency paths. In *EMNLP'15*. 1785–1794.
[36] Rui Yan, Mirella Lapata, and Xiaoming Li. 2012. Tweet Recommendation with Graph Co-ranking. In *ACL '12*. 516–525.
[37] 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.
[38] Rui Yan, Yiping Song, Xiangyang Zhou, and Hua Wu. 2016. "Shall I Be Your Chat Companion?" Towards an Online Human-Computer Conversation System. In *CIKM'16*. 649–658.
[39] Rui Yan, Dongyan Zhao, et al. 2017. Joint Learning of Response Ranking and Next Utterance Suggestion in Human-Computer Conversation System. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 685–694.
[40] Zi Yang, Keke Cai, Jie Tang, Li Zhang, Zhong Su, and Juanzi Li. 2011. Social Context Summarization. In *SIGIR '11*. 255–264.
[41] Lili Yao, Yaoyuan Zhang, Yansong Feng, Dongyan Zhao, and Rui Yan. 2017. Towards Implicit Content-Introducing for Generative Short-Text Conversation Systems. In *Proceedings of EMNLP 2017*. 2190–2199.
[42] Jonathan S Yedidia, William T Freeman, Yair Weiss, et al. 2000. Generalized belief propagation. In *NIPS*, Vol. 13. 689–695.
[43] Biao Zhang, Jinsong Su, Deyi Xiong, Yaojie Lu, Hong Duan, and Junfeng Yao. 2015. Shallow Convolutional Neural Network for Implicit Discourse Relation Recognition. In *EMNLP*. 2230–2235.
[44] Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. 2011. Comparing twitter and traditional media using topic models. In *ECIR '11*. 338–349.
[45] Xiangyang Zhou, Daxiang Dong, Hua Wu, Shiqi Zhao, Dianhai Yu, Hao Tian, Xuan Liu, and Rui Yan. 2016. Multi-view Response Selection for Human-Computer Conversation.. In *EMNLP'16*. 372–381.