Learning to Respond with Deep Neural Networks for Retrieval-Based Human-Computer Conversation System

Rui Yan
Baidu Inc.
No. 10, Xibeiwang East Road,
Beijing 100193, China
yanrui02@baidu.com

Yiping Song
Baidu Inc.
No. 10, Xibeiwang East Road,
Beijing 100193, China
songyiping01@baidu.com

Hua Wu
Baidu Inc.
No. 10, Xibeiwang East Road,
Beijing 100193, China
wu_hua@baidu.com

ABSTRACT

To establish an automatic conversation system between humans and computers is regarded as one of the most hardcore problems in computer science, which involves interdisciplinary techniques in information retrieval, natural language processing, artificial intelligence, etc. The challenges lie in how to respond so as to maintain a relevant and continuous conversation with humans. Along with the prosperity of Web 2.0, we are now able to collect extremely massive conversational data, which are publicly available. It casts a great opportunity to launch automatic conversation systems. Owing to the diversity of Web resources, a retrieval-based conversation system will be able to find at least some responses from the massive repository for any user inputs. Given a human issued message, i.e., query, our system would provide a reply after adequate training and learning of how to respond. In this paper, we propose a retrieval-based conversation system with the *deep learning-to-respond* schema through a deep neural network framework driven by web data. The proposed model is general and unified for different conversation scenarios in open domain. We incorporate the impact of multiple data inputs, and formulate various features and factors with optimization into the deep learning framework. In the experiments, we investigate the effectiveness of the proposed deep neural network structures with better combinations of all different evidence. We demonstrate significant performance improvement against a series of standard and state-of-art baselines in terms of p@1, MAP, nDCG, and MRR for conversational purposes.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; I.5.1 [**Pattern Recognition**]: Models—*Deep learning*

Keywords

Learning-to-respond; conversation system; contextual modeling; deep neural networks

SIGIR '16, July 17-21, 2016, Pisa, Italy

DOI: http://dx.doi.org/10.1145/2911451.2911542

1. INTRODUCTION

To have a virtual assistant and/or chat companion system in open domains with adequate artificial intelligence has seemed illusive, and might only exist in Sci-Fi movies for a long time. Recently, the goal of creating an automatic human-computer conversation system, as our personal assistant or chat companion, is no longer an illusion far away. Due to easily accessible "big data" for conversations on the Web, we might be able to learn how to respond and what to respond given (almost) any inputs. It is likely to be a great timing to build data-driven, open-domain conversation systems between humans and computers.

Building conversation systems, in fact, has attracted much attention over the past decades. In early years, researchers have investigated into task-oriented conversation systems [44, 33, 36], 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 replies to handle the particular task [23]. 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 limitation 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 [2, 26, 9, 35, 17, 31, 6]. 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 beyond any doubt [34].

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[1], Twitter[2]) and community question answering (cQA) platforms (e.g., Baidu Zhidao[3], Yahoo! Answers[4]). These resources provide a unique opportunity to build collections of naturally occurring conversations that are orders of magnitude larger

[1]http://www.facebook.com
[2]http://www.twitter.com
[3]http://www.zhidao.baidu.com
[4]https://answers.yahoo.com

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, and return a sensible response.

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 ought to identify appropriate candidate replies from a very large volume of data. Besides, the proposed model should also be general and unified for different conversation scenarios. In a conversation system, usually there is additional information to use such as "contexts" (a.k.a. previous utterance sentences in a continuous conversation session). Therefore, capturing and integrating as much information as possible in a proper way is important for conversation systems.

In this paper, we propose a "deep learning-to-respond" framework for open-domain conversation systems. We create a huge conversational dataset from Web, and the crawled data are stored as an atomic unit of natural conversations: an utterance, namely a *posting*, and its *reply*. Each ⟨*posting-reply*⟩ can be regarded as a single-turn conversation. For a given query, we first apply traditional keyword-based retrieval methods and obtain a list of candidate replies; each reply is associated with its antecedent posting. We then enhance the current query by adding its contexts, i.e., one or more previous utterances in the current conversation session. Thus, we obtain a set of reformulated queries as well as the original query. A deep neural network (DNN)-based ranker thereafter tells how each candidate reply/post is related to a (reformulated) query, and yields a ranking list for each (reformulated) query. We merge the ranking lists corresponding to different reformulations. In this way, we are able to organically incorporate into the conversation system multi-dimension of ranking evidences including queries, contexts, candidate postings and/or replies, which is a novel insight.

The proposed reformulation approach and merging strategy provide a new means of conversation modeling, especially multi-turn conversations. By using previous utterances, we are aware of background information of the query, which might be informative. Moreover, different reformulations can capture different aspects of background information; their resulting ranked lists are further merged by a novel formula, in which we consider the relatedness between the reformulated queries (with context) and the original one.

The DNN ranker, serving as the core of "deep learning-to-rank" schema, models the relation between two sentences (query versus context/posting/reply). We use a bi-directional recurrent neural network to propagate information across words; a convolutional neural network layer further captures patterns of adjacent words. Then a matching layer combines the information in each individual sentence. Note that our DNN is a generic framework and applies to *Query-Reply*, *Query-Posting* and *Query-Context* in a unified way.

We conduct extensive experiments in a variety of 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 human conversation resources. We run experiments against several other rival algorithms to verify the effectiveness of the proposed DNN model. Our system outperforms standard and state-of-the-art baselines regarding a variety of evaluation metrics in terms of p@1, MAP, nDCG and MRR metrics. The result indicates that our conversation system is rather helpful to facilitate conversations between human and computer.

To sum up, our contributions are mainly as follows:

- We propose a "deep learning-to-respond" schema for automatic human-computer conversation systems with deep neural networks (DNNs). The deep learning model is general and well unified to adapt for various conversation scenarios.
- We propose a novel concept to model *context* in a continuous conversation session in multi-turns. The proposed query reformulation can capture different aspects of background information. We design an insightful framework to merge the ranking lists of candidates to all (reformulated) queries.
- We propose a new response ranking paradigm give each (reformulated) query, incorporating multi-dimension of ranking evidences: *Query-Reply*, *Query-Posting* and *Query-Context*, which is an adaption for conversational scenarios.

The rest of the paper is organized as follows. We start by reviewing related work. In Section 3, we describe the task modeling and proposed framework for conversation systems. In Sections 4 and 5, we introduce the detailed mechanisms of contextual query reformulation and the deep learning-to-respond architecture. We devise experimental setups and evaluations against a variety of baselines and discuss results in Section 6. Finally we draw conclusions in Section 7.

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 [33, 36]. These rule-based approaches requires no data or little data for training, while instead require much manual effort 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 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 [31, 6].

From specific domains to open domain, the need for a huge 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.* 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 [12], but have a major bottleneck of the creation of the knowledge base (i.e., question-answer pairs) [13]. Researchers propose to augment the knowledge base with question-answer pairs derived from plain texts [24, 3]. The number of resource pairs can be, to some extent, expanded, but are still relatively small while the performance is not quite stable either.

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 [35]. A series of information retrieval-based methods are applied to short text conversation using microblog data [9, 14, 17, 16]. Higashinaka *et al.* also combine template generation with the search-based methods [6]. 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 [26]. 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.

Unlike previous work, we conduct a novel study of retrieval-based automatic conversation systems with a *deep learning-to-respond* schema via deep learning paradigm. We formulate the possible factors into a deep neural network architecture and further investigate the potential of combining different ranking evidences for the candidate responses. Deep learning structures are well formulated to describe instinct semantic representations.

2.2 Deep Neural Networks

In recent years, deep neural networks (DNNs, also known as *deep learning*) have made significant improvement in NLP [11]. DNNs are highly automated learning machines; they can extract underlying abstract features of data automatically by exploring multiple layers of non-linear transformation [1].

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* [19]. Each dimension in the vector captures some (anonymous) 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 [10], whereas RNNs keep one or a few hidden states, and collect information along the word sequence in an iterative fashion [38, 37, 39]. Socher *et al.* leverage sentence parse trees and build recursive networks [30]. Mou et al. [21, 20] propose syntax-aware convolution based on parse trees. However, conversational utterances are usually casual, and hence recursive models are less applicable in conversation systems. We prefer structure-insensitive models like CNNs and RNNs.

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 [5], discourse unit recognition [45], textual entailment recognition [27], 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 [5, 28]. Hu *et al.* develop word-by-word matching approaches [7], and obtain a similarity matrix between two sentences. Very recently, Rocktäschel *et al.* propose context-aware matching approaches [27], 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, especially with multiple query reformulations in our scenario. For efficiency consideration, we leverage vector concatenation, which is simple yet effective.

Although the studies of sentence-pair modeling described above are similar to our DNN to some extent, the proposed learning-to-rank model is more than traditional ranking or matching. We have multiple query reformulations with "contexts". After computing the similarity between the query and reply/post/context, our DNN further merges the ranking results corresponding to different reformulated queries.

3. TASK MODELING

In this section, we provide a big picture of the proposed *learning-to-respond* schema for conversation systems. We illustrate the task modeling for conversations, and establish the pipeline with several processing procedures including data collection, search and retrieval, contextual query reformulation, DNN-based ranking with scoring, and ranked list fusion. We briefly go over through the pipeline and elaborate the details in the next section.

Table 1: An example of the original microblog $posting$ and the associated $replies$. Each posting might have more than one reply, e.g., $Reply_1$ and $Reply_2$. To create our database of conversation data, we separate different replies to a same post, and obtain ⟨*post-reply*⟩ pairs. We store two *Posting-Reply* pairs in the conversational dataset, i.e., ⟨$Posting$-$Reply_1$ ⟩ and ⟨$Posting$-$Reply_2$ ⟩. User accounts are anonymized.

$Posting$: 近视了需要戴眼镜...   (I need a pair of glasses because of the myopia...)
$Reply_1$: 我送你眼镜！   (I will offer the glasses for you!) $Reply_2$: 可以恢复的，别紧张 . . .   (You will be recovered. Don't worry.)

Table 2: Part (I) indicates a real human (denoted by A) - computer (denoted by B) conversation scenario, while Part (II) indicates our proposed task modeling and formulations. A_2 is the current user-issued query. We have contexts and reformulated queries as listed. '⊞' is the literal concatenation action. Note that the selected response $Reply_1$ is associated with a $Posting$ in the conversational database shown in Table 1.

(I)

Human-Computer Conversation
A_1: 天哪一把年纪的人居然近视了   (OMG I got myopia at such an "old" age) B_1: 真的吗？   (Really?) A_2:嗯哪。求个眼镜做礼物！   (Yeah. Wish a pair of glasses as a gift.)
B_2:我送你眼镜！   (I will offer the glasses for you!)

(II)

Task Formulation
User query: $q_0 = A_2$ **Context information:** $\mathbb{C}$=$\{c_1$=A_1,c_2=$B_1\}$ **Reformulated queries:** q_1=$A_2 \boxplus A_1$, q_2=$A_2 \boxplus B_1$ q_3=$A_2 \boxplus A_1 \boxplus B_1, \ldots$
Top-1 ranked response: $r^\star = Reply_1$

Data collection. With the prosperity of Web 2.0, people interactively communicate with each other on the Web, which provides a huge thesaurus for conversation data. We collect a large amount of data samples from social media such as microblog websites, forums, cQA bases, etc. Users can publish a posting message visible to the public, and then receive one or more replies or comments in response to their posting. The communication can have a single turn as well as multiple turns. We illustrate an example in Table 1. Due to the heterogeneity of the sources, we treat each utterance, in multi-turn conversations, with its subsequent one as a posting-reply pair (i.e., our database is context-free). For a posting with multiply replies, we separate them and construct different $\langle p, r \rangle$ pairs. Table 1 shows the pre-processed samples in our dataset, applied to a real human-computer conversation illustrated in Table 2. In the sample shown in Table 1, the first message of a conversation is typically unique. There are many flexible forms to "respond" to a given message, which is exactly the nature of real conversations: various responses are all possibly appropriate, with different aspects of information to fulfill a conversation. We separate the posting and replies as a group of ⟨*posting-reply*⟩ pairs. The data repository is demonstrated to be a rich resource to facilitate human-computer conversations.

Search and retrieval. In the scenario of conversations, the user issues a query (q_0 in Table 2), which may be a sentence or a few terms. We apply a standard retrieval process via keyword search using traditional *tf.idf* weighting schema [18] on the conversation data-base (formatted as an inverted index prepared off-line) based

on the light-weight search provided by Baidu[5]. Note that we treat each pair of posting and reply as a short *virtual document*, which is not a traditional process. In this way, the retrieved "virtual document" comprises two parts: the *candidate reply*, namely r, along with its antecedent *posting*, namely p.

Contextual query reformulation. A single query may not fully convey user intentions in (multi-turn) conversations, as illustrated in Table 2. Under such conditions, we usually have context information to use. We propose a novel insight to model the conversation task. In particular, we view previous utterances from both sides as *contexts*, denoted as $\mathbb{C}=\{c_i\}$. One or more utterances in $\mathbb{C}$ can be used to enhance q_0 so as to provide more information. We call this a *contextual query reformulation* process. Moreover, the context may comprise several sentences, and hence we have several strategies to reformulate the original query. Each reformulated query is denoted as q_i. More details will be given in Section 4.

DNN-based scoring, ranking, and ranked list fusion. We apply a deep neural network (DNN)-based model to rank optimization. In particular, we design a bi-directional long short term memory (LSTM) neural network to capture sentence-level semantics of a query q_0, candidate reply and the associated posting, i.e., $\langle p, r\rangle$, as well as context $\mathbb{C}$. A matching layer combines multi-dimensions of the sentence information, so that we know how candidate replies with associated postings are related to the query and contexts. Analogous to the traditional *Query-Document* matching, the relevance ranking can be measured via *Query-Reply* matching, the additional *Query-Posting* matching, as well as *Query-Context* matching. Intuitively, when a query and a posting look similar, they might share the same response (Tables 1–2). For each (reformulated) query, DNN ranks candidates replies with relevance scores from *Query-Query* and *Query-Posting*. We further merge all candidate ranked lists of all contextual reformulations using weighted fusion. The weight is controlled by the relevance between the original query and the reformulated one with contextual information, i.e., *Query-Context*. For continuous conversations, contexts can be used to optimize the response selection for the given query.

Table 3 summarizes the input and output of the proposed system with deep learning-to-respond schema. Given a query with context, the proposed model would return a response—which has the highest overall (merged) ranking score $\mathcal{F}(.)$—from the pre-constructed repository. We use DNN to assess the relevance between candidate replies, postings and reformulated queries with different combinations of contexts. The DNN also merges the ranking scores corresponding to different contextual query reformulations.

We apply hinge loss with negative sampling to train the network. Gradients can be back-propagated all the way back from merging, ranking, sentence pairing, to individual sentence modeling. Therefore, all these heterogeneous ranking evidences are integrated together through the proposed Deep Learning-to-Respond schema.

Table 3: Symbols and annotations for problem formulation.

q_0	the current query
r, p	candidate reply with the associated antecedent posting
$\mathbb{C}=\{c_i\}$	contexts (utterances before q_0). c_i is a utterance in $\mathbb{C}$
$\mathcal{Q}=\{q_i\}$	reformulated query: q_0 concatenates with some c_i
$f(.)$	matching metric for *Query-Reply*
$g(.)$	matching metric for *Query-Posting*
$h(.)$	matching metric for *Query-Context*
Input	Conversation repository: $\{\langle p, r\rangle\}$ Query: q_0 Context: $\mathbb{C}$
Output	Selected response: $r^* = \underset{r}{\operatorname{argmax}}\, \mathcal{F}\big(r \mid q_0, \mathbb{C}, \{\langle p, r\rangle\}\big)$

4. CONTEXTUAL REFORMULATION

Generally, context information may be informative (but sometimes may be not) when modeling a query. It is non-trivial to explore different strategies to utilize context information for conversations. In this section, we describe the proposed contextual query reformulation approach.

The contextual query reformulation strategies are mainly inspired by the following observations:

- Some context utterances are informative, while others are not. As the example in Table 2 illustrates, the context utterance "*Really?*" should not be considered as informative as the context utterance "*OMG I got myopia at such an 'old' age*". Therefore, we shall have different reformulations: we can add the contexts as a whole, or we can add the contexts one-by-one. The intuition is that context sentences are not equally informative, and they shall be used differently.

- Given a context of N sentences, the number of possible ways to concatenate the original query is theoretically 2^N. As a combinatory problem, the number of combinations will grow exponentially as N grows up. Hence we need to impose constraints on the contextual query reformulation strategies. An acceptable solution should be (at most) linear as N grows. We ought to use a subset of reformulations, and combine such strategies to approximate all possible reformulations. The results are then merged by DNN with different weights, analogous to Bayesian model average over different reformulations.

- Under the scenario of a continuous conversation, we observe that contexts, no matter from the *human* side or from the *computer* side, play a similar role to the future conversation. Therefore, we do not distinguish utterances from two sides in the context. Nevertheless, how relevant is a context with the current query is assessed by the *Query-Context* matching function, and the degree of relevance matters when DNN merges different ranked lists from reformulated queries.

Without loss of generosity, let us assume there are N sentences in the context $\mathbb{C}$, i.e., $c_1, \cdots, c_N \in \mathbb{C}$ being previous utterances in the current conversation session. We add one or more context sentences to the query q_0 and obtain a set of reformulated queries (each is denoted as $q_i \in \mathcal{Q}$). To reduce the explosive number of all possible 2^N combinations, we restrict the contextual query reformulation strategies in practice as follows:

- *No Context*: the simplest reformulation strategy is that no context information will be added, i.e., $\mathcal{Q}_{\text{No Context}} = \{q_0\}$.

- *Whole Context*: we do not distinguish different context sentences and hence incorporate the entire contexts as a whole, i.e., $\mathcal{Q}_{\text{Whole Context}} = \{q_0, q_0 \boxplus \mathbb{C}\}$.

- *Add-One*: we concatenate q_0 with one context sentence, one at a time, i.e., $\mathcal{Q}_{\text{Add-One}} = \{q_0, q_0 \boxplus c_1, \ldots, q_0 \boxplus c_N\}$.

- *Drop-Out*: we concatenate q_0 with the whole context while leave-one-out each context sentence, one at a time, i.e., $\mathcal{Q}_{\text{Drop-Out}} = \{q_0, q_0 \boxplus [\mathbb{C} \backslash c_1], \ldots, q_0 \boxplus [\mathbb{C} \backslash c_N]\}$.

[5]Baidu is the largest Chinese search engine provider. Some of the services are available at http://www.baidu.com.

- *Combined*: the combination of all strategies, i.e., $\mathcal{Q} = \mathcal{Q}_{\text{No Context}} \bigcup \mathcal{Q}_{\text{Whole Context}} \bigcup \mathcal{Q}_{\text{Add-One}} \bigcup \mathcal{Q}_{\text{Drop-Out}}$.

'$\boxplus$' refers to literal concatenation, where the order of context and the query is preserved. "$\mathbb{C}\backslash c_i$" indicates c_i is filtered out from $\mathbb{C}$. The benefits for all the contextual query reformulation strategies are that the cost is fixed (for $\mathcal{Q}_{\text{No Context}}$ and $\mathcal{Q}_{\text{Whole Context}}$) or linear (for $\mathcal{Q}_{\text{Add-One}}$ and $\mathcal{Q}_{\text{Drop-Out}}$) along as N grows. The intuition for *Add-One* and *Drop-Out* strategies is based on a finer-granularity: to incorporate relevant context sentences only, or to exempt one irrelevant context sentences. The last strategy combines all these strategies to approximate all possibilities.

5. DEEP LEARNING TO RESPOND

In this section, we describe the deep model for query-reply ranking and merging. Our model first determines the score of a candidate reply given the (reformulated) query, based on the candidate reply and its associated posting (Subsection 5.1). Then the model merges the score by summing over all query reformulations including the original query with a *gating* (product) mechanism (Subsection 5.2). In this way, all heterogeneous ranking evidence is combined organically in a differentiable manner.

5.1 Sentence Pair Modeling

As mentioned, our model first determines the relationship for *Query-Reply*, *Query-Posting*, and *Query-Context* matchings. The three scoring functions are defined as:

- $f(q, r)$: This scoring function directly judges the relatedness between a reply and the (reformulated) query. A larger score achieved means more relevance of the candidate reply.
- $g(q, p)$: If the associated posting of the reply is similar to the query, its subsequent reply is likely to be an appropriate response.
- $h(q, q_0)$: This scoring function tells how the reformulated query is correlated with the original q_0. A more relevant context, i.e., a larger $h(q, q_0)$, should leads to a more confident ranking result, and the scores from $f(q, r)$ and $g(q, p)$ will be credited with more weights for the final Sum fusion.

The scoring function $f(q, r)$ outputs a scalar in $\mathbb{R}$ (appropriateness or inappropriateness) in for a particular candidate reply, while the latter two functions serving as an adjustment or "*gate*", which are squashed by a logistic function into the $(0, 1)$ range. Nevertheless, all the above functions are computed by the same deep neural network architecture (except for the last activation function), but their parameters are different so that the three scoring functions can depict different meanings. In particular, the deep structure for sentence pair modeling includes the following components.

5.1.1 Word Embeddings

Traditional models usually treat a word as a discrete token; thus, the internal relation between similar words would be lost. Word embeddings [19] are 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 [19].

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.

5.1.2 Bi-Directional LSTM

We use a bi-directional long short term memory (Bi-LSTM) recurrent network to propagate information along the word sequence.

As reviewed in Section 2, 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 further using memory cells and gates to learn long term dependencies within a sequence [32, 25]. 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 the word embedding at position t in the sentence. LSTM outputs a representation h_t for position t, given by

$$\begin{bmatrix} i_t \\ f_t \\ o_t \\ l_t \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{bmatrix} W \cdot \begin{bmatrix} h_{t-1} \\ e_t \end{bmatrix} \tag{1}$$

$$\tilde{h}_t = f_t \cdot \tilde{h}_{t-1} + i_t \cdot l_t$$

$$h_t^s = o_t \cdot \tilde{h}_t$$

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 blind of future words. The variant Bi-LSTM [4] 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., $h_t = \left[\overrightarrow{h_t}; \overleftarrow{h_t}\right]$.

5.1.3 Convolution

We further apply a convolutional neural network (CNN) to extract local neighboring features of successive words—i.e., discriminative word sequences can be detected—yielding a more composite representation of the sentences. The structure of CNN in this work is similar to [10], shown in Figure 1. Unlike RNNs, CNNs only impose local interactions between successive words within a filter (size m).

Concretely, we build a CNN upon the output of Bi-LSTM. For every window with the size of m in Bi-LSTM output vectors, i.e., $(H_t)_m = [h_t, h_{t+1}, \cdots, h_{t+m-1}]$, where t is a certain position, the convolutional filter $F = [F(0), \ldots, F(m-1)]$ will generate a vector sequence using the convolution operation "$*$" between the two vectors. More formally, the convolution results in a vector where each component is as follows:

$$o_F = \tanh\left[\sum_{i=0}^{m-1} h(t+i) * F(i)\right] \tag{2}$$

In practice, we also add a scalar bias b to the result of convolution. In this way, we obtain the vector o_F is a vector, each dimension corresponding to each word in the sentence.

Notice that the above equation describes a single "slice" of convolution. In fact, we may have multiple feature filters and thus

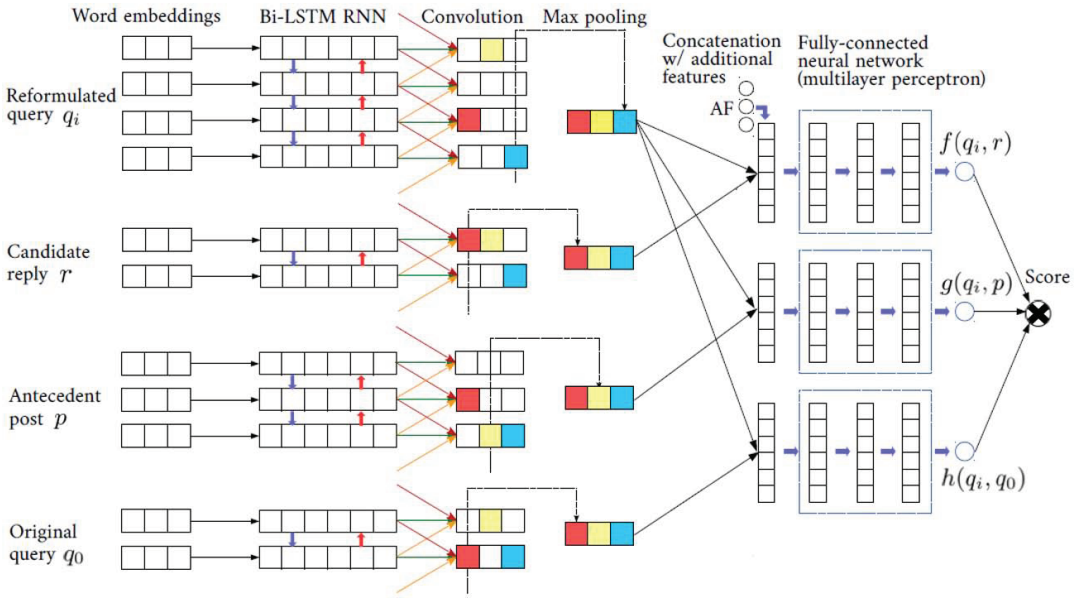

Figure 1: The deep neural network architecture for matching sentence pairs.

multiple feature maps. Different filters do not share parameters (F and b), so that they can capture different meanings.

5.1.4 *Pooling, Concatenation, and Matching*

On the basis of sentence representations using Bi-LSTM with CNN, we can model the interactions between two sentences. We apply pooling to aggregate information along the word sequence. In particular, a max pooling layer chooses the maximum value in each dimension in the feature maps after the convolution, indicating how much the feature is most satisfied along the sequence.

We concatenate two individual sentences' vector representations (with possible additional features), which are then fed to an ensuing network for further information mixing. Vector concatenation for sentence matching is also applied in other studies like [45, 22], which is effective yet of low complexity order, compared with other word-by-word matching [7], or attention methods [27].

The joint vector is then passed through a 3-layer, fully-connected, feed-forward neural network, also known as *multi-layer perception* (MLP) [1], which allows rich interactions between a sentence pair from one of the three components. The network enables to extract features automatically, starting from lower-level representations to higher-level ones.

Finally, a single neuron outputs the matching score of two sentences. As mentioned, $f(q, r)$ is in $\mathbb{R}$; hence the final scoring neuron is essentially a linear regression. For $g(q, p), h(q, q_0) \in (0, 1)$, we apply a sigmoid/logistic function given by $\sigma(\cdot) = \frac{1}{1+e^{-\cdot}}$.

5.2 Merging

In the previous subsection, we have described how the model captures the relation among sentence pairs. Now, we merge the scores of a particular candidate r in terms of different contextual query reformulations. As discussed in Section 5.1, if a posting is more related to the (reformulated) query and the reformulated query is more related to the original query, then $f(q, r)$ would be more reliable. Inspired by this observation, $g(q, p)$ and $h(q, q_0)$ as designed as two adjusting "gates". In particular, scores from different query reformulations are summed, weighted by these two gates. The overall ranking score of a candidate reply r and q_0 is defined as follows. The equation and spirit also appear similar to the integration of a $sum\text{-}product$ process, which combines the sum operations and $product$ operations:

$$\mathcal{F}(q_0, r) = \sum_{i=0}^{|\mathcal{Q}|} \Big(h(q_0, q_i) \sum_{p} \big(f(q_i, r) \cdot g(q_i, p) \big) \Big) \quad (3)$$

Here we propose to sum over all postings associated with the reply. Different data repository will have different settings: a candidate reply is possible to associate with more than one postings. In our data settings, each reply is associated with only one posting. However, Equation (3) is general and extensible.

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 $\mathcal{F}(q_0, r^+)$ 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 $\mathcal{F}(q_0, r^+)$ to be as least $\mathcal{F}(q_0, r^-)$ plus a margin Δ. Thus, the training objective is to

$$\underset{\Omega}{\text{minimize}} \sum_{q_0, r^+} \max \left\{ 0, \Delta + \mathcal{F}(q_0, r^+) - \mathcal{F}(q_0, r^-) \right\} + \lambda \|\Omega\|_2^2 \quad (4)$$

where we add an ℓ_2 penalty with coefficient λ for all the parameters $\Omega = \{\theta; \eta; \phi\}$ which are weight and bias values optimized by the network from multi-dimensions of ranking evidences, i.e., *Query-Reply*, *Query-Posting* and *Query-Context*, correspondingly.

5.3 Training

As our model is (almost) everywhere differentiable, the parameters of the networks are optimized with stochastic gradient descent using the back prorogation algorithm to compute the gradients. The gradients can be propagated all the way back through merging, gating, matching, and individual sentence modeling. In this way, heterogeneous information (ranking evidences) can be incorporated organically with our model under the unified deep architecture. Accordingly to the objective function to optimize $\mathcal{O}(.)$ in Equation (4), it is sufficient to learn the model by computing the gradients with respect to the model parameters θ, η and ϕ; that is, our goals

Table 4: Data statistics. Postings and replies are all unique.

Source	#Posting	#Reply	#Vocabulary
Zhidao	8,915,694	3,705,302	1,499,691
Douban	10,618,981	2,963,226	483,846
Tieba	4,189,160	3,730,248	1,046,130
Weibo	186,963	393,654	119,163
Misc.	3,056	1,548	4,297
Total	9,023,854	7,293,978	2,857,378

are to compute $\frac{\partial \mathcal{O}}{\partial \theta}$, $\frac{\partial \mathcal{O}}{\partial \eta}$ and $\frac{\partial \mathcal{O}}{\partial \phi}$ for *Query-Reply*, *Query-Posting* and *Query-Context*.

6. EXPERIMENTS AND EVALUATION

In this section, we evaluate our model for conversation task against a series of baselines based on a huge conversation resource. The objectives of our experiments are to 1) evaluate the effectiveness of our proposed deep learning-to-respond schema, and 2) evaluate contextual reformulation strategies and components of multi-dimension of ranking evidences for the conversational task.

6.1 Dataset

As mentioned, we collected massive conversation resources from various forums, microblog websites, and cQA platforms including Baidu Zhidao[6], Douban forum[7], Baidu Tieba[8], Sina Weibo[9], etc. We conducted data filtering and cleaning procedures by removing extremely short replies and those of low linguistic quality such as meaningless babblings according to the evaluation framework put forward in [40, 42], so as to maintain a meaningful, high-quality conversation record. In total, the database contains $\sim$10 million $\langle posting, reply \rangle$ pairs. Some statistics are summarized in Table 4.

We constructed the dataset of 1,606,583 samples to train the deep neural networks, 357,018 for validation, and 11,097 for testing. It is important that the dataset for learning does not overlap with the database for retrieval, so that we strictly comply with the machine learning regime. For each training and validation sample, we randomly chose a reply as a negative sample. Validation was based on the accuracy of positive/negative classification. For the test set, we hired workers on a crowdsourcing platform to judge the appropriateness of 30 candidate replies retrieved for each query. Each sample was judged by 7 annotators via majority voting based on the *appropriateness* for the response given the query and contexts (if any): "1" denotes an appropriate response and "0" indicates an inappropriate one.

6.2 Experimental Setups

6.2.1 Hyperparameters

In our proposed model, we used 128-dimensional word embeddings, and they were initialized randomly and learned during training. As our dataset is in Chinese, we performed standard Chinese word segmentation. We maintained a vocabulary of 177,044 phrases by choosing those with more than 2 occurrences.

The bi-directional LSTM has 128 hidden units for each dimension; CNN is 256 dimensional with a window size of 3. We used stochastic gradient descent (with a mini-batch size of 100) for optimization, gradient computed by standard back-propagation. Initial learning rate was set to 0.8, and a multiplicative learning rate decay was applied. The above parameters were chosen empirically. We used the validation set for early stopping.

[6]http://www.zhidao.baidu.com

[7]http://www.douban.com

[8]http://www.tieba.baidu.com

[9]http://www.weibo.com

6.2.2 Evaluation Metrics

Given the ranking lists (annotated by crowdsourced workers) for test queries, we evaluated the performance in terms of the following metrics: precision@1 (p@1), mean average precision (MAP) [31, 43], and normalized discounted cumulative gain (nDCG) [8, 41]. 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 also provided the 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 responses as candidates. We aimed at selecting as many appropriate responses as possible into the top-k list and rewarding methods that return suitable replies on the top.

Formally, the metrics are computed as follows.

$$\text{nDCG@i} = \frac{1}{|\mathcal{T}|} \sum_{\text{q} \in \mathcal{T}} \frac{1}{\text{Z}} \sum_{i=1}^{k} \frac{2^{r_i} - 1}{\log(1+i)}$$

where $\mathcal{T}$ indicates the testing query set, k denotes the top-k position in the ranking list, and Z is a normalization factor obtained from a perfect ranking. r_i is the relevance score for the i-th candidate reply in the ranking list (i.e., 1: appropriate, 0: inappropriate).

MAP is computed by

$$\text{MAP} = \frac{1}{|\mathcal{T}|} \sum_{\text{q} \in \mathcal{T}} \frac{1}{N_q} \sum_{i=1}^{k} P_i \times r_i$$

Here N_q is the number of appropriate responses selected, and P_i is the precision at i-th position for the query.

Since we use real conversations for testing, we also have the human response taken from the human-human conversation session as one of candidate replies ordered in the ranked list. Hence we include the Mean Reciprocal Rank (MRR) evaluation computed as:

$$\text{MRR} = \frac{1}{|\mathcal{T}|} \sum_{\text{q} \in \mathcal{T}} \frac{1}{rank(q)}$$

where $rank(q)$ is the position of the original response in the candidate ranking list. Unlike MAP and nDCG, which examine the ranks of all appropriate responses, MRR focuses on evaluating the capability of retrieval systems to find (perhaps) the best response. MRR is useful but does not test the full capability because there can be more than one appropriate responses to fulfill a conversation.

6.2.3 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 pre-processing 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 one sentence in the source language to a sentence in the target language. If we treat queries and replies as separate languages, we can train a translation model to

Table 5: Retrieval performance against baselines with our proposed adaption of contextual reformulation. '⋆' indicates that we accept the improvement hypothesis of DL2R over the best baseline by Wilcoxon test at a significance level of 0.01. Performance of both generative methods and retrieval methods. For generative methods, they generate one response given each query. Hence the p@1 in fact refers to accuracy. Other metrics are not applicable.

Model	p@1	MAP	nDCG@5	nDCG@10	nDCG@20	MRR
SMT (Ritter et al., [26])	0.363					
LSTM-RNN (Sutskever et al., [32])	0.441					
NRM (Shang et al., [29])	0.465					
Random Match	0.266	0.246	0.247	0.289	0.353	0.083
Okapi BM25	0.272	0.253	0.337	0.302	0.368	0.169
DeepMatch (Lu and Li, [17])	0.457	0.317	0.419	0.454	0.508	0.275
LSTM-RNN (Palangi et al., [25])	0.338	0.283	0.330	0.371	0.431	0.228
ARC (Hu et al., [7])	0.394	0.294	0.397	0.421	0.477	0.232
DeepMatch w/ context adaption	0.603	0.378	0.555	0.584	0.628	**0.349**
LSTM-RNN w/ context adaption	0.362	0.296	0.354	0.395	0.453	0.237
ARC w/ context adaption	0.400	0.309	0.383	0.422	0.480	0.319
Deep Learning-to-Respond (DL2R)	**0.731***	**0.416***	**0.663***	**0.682***	**0.717***	0.333

Table 6: Performance evaluations of different contextual query reformulation strategies.

	p@1	MAP	nDCG@5	nDCG@10	nDCG@20	MRR
No Context	0.522	0.340	0.476	0.509	0.559	0.296
Whole Context	0.698	0.404	0.635	0.657	0.696	0.327
Add-One	0.716	0.411	0.650	0.670	0.706	0.322
Drop-Out	0.720	0.413	0.656	0.675	0.711	0.328
Combined	0.731	0.416	0.663	0.682	0.717	0.333

Table 7: Performance evaluations of different components with multi-dimension of ranking evidences.

	p@1	MAP	nDCG@5	nDCG@10	nDCG@20	MRR
Query-Reply w/o Query-Context	0.522	0.340	0.476	0.509	0.559	0.296
Query-Posting w/o Query-Context	0.510	0.302	0.404	0.425	0.489	0.285
Query-Reply w/ Query-Context	0.596	0.366	0.528	0.561	0.603	0.327
Query-Posting w/ Query-Context	0.563	0.362	0.483	0.516	0.568	0.316
Full Combination	0.731	0.416	0.663	0.682	0.717	0.333

"translate" queries into replies. We implemented the phrase-based translation idea for conversation proposed in [26].

• *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) [32]; 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 with a global-local attention schema.

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. Since our approach is retrieval-based, we select strong retrieval-based methods to make a thorough comparison.

• *Random Match.* The method randomly selects replies from the retrieved list for each query. Be aware it is not *true* random because it only randomizes the order of the retrieved results. The true random match is too weak to be included as a decent baseline.

• *Okapi BM25.* We include the standard retrieval technique to rank candidate replies. For each query, we retrieve the most relevant reply using BM25 model [18] from the corpus.

• *DeepMatch.* The DeepMatch method considers multiple granularity from the perspective of topics, obtained via LDA [17].

• *ARC.* The ARC approach is a CNN based method with convolutionary layers which construct sentence representations and produce the final matching scores via a MLP layer [7].

• *Deep Learning-to-Respond (DL2R).* We propose the DL2R system based on three novel insights: 1) the integration of multi-dimension of ranking evidences, 2) context-based query reformulations with ranked lists fusion, and 3) deep learning framework for the conversational task. There is actually a series of variants of DL2R model with different components and different context utilization strategies. We will first report the performance comparisons between DL2R against baselines and then show the details of components and strategies in Section 6.4.

6.3 Overall Performance

We compare the performance of all methods including baselines and our proposed DL2R model measured in terms of p@1, MAP, nDCG and MRR. In Table 5 we list the overall results against al-

l baseline methods. Our proposed method DL2R shows clearly better performance than the baseline methods. On average, DL2R achieves an average +38.63% improvement (averaged on all metrics) compared with the strongest baseline group with context adaption (in Table 5). We then discuss the comparisons in details.

We illustrate the result from generative methods. Given one user query, the generative methods generally provide one generation as the response to output. Hence we do not compare MAP or nDCG@1 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, while LSTM-RNN and NRM perform better than the SMT method. The reasons can be ascribed to two aspects: firstly, SMT is not instinctively tailored for conversation systems and secondly, deep neural networks for LSTM-RNN and NRM will be more likely to learn a better representation for the queries and then return a better decoded generation as response. The generated responses are in general quite ambiguous or broad to answer a wide range of queries, but not specific enough. Such responses might be relevant but not appropriate enough to make a meaningful conversation, which have been raised as a problem in [15, 29].

We can see great improvement for DL2R against original retrieval-based baselines. *Random Match* is a lower bound for all baselines. As we mentioned, it randomizes the order of the retrieved results. Hence the result is comparable to that of BM25 but slightly worse. *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, which is not surprising. Deep learning systems are proved to have strong capabilities to learn the abstractive representation [1, 10, 30], while BM25 only utilizes the shallow representation of term-level retrieval. The deep learning algorithm groups clearly overwhelm shallow learning algorithms, yet it is interesting to see that DL2R still outperforms the other deep learning baselines in Table 5. The benefits might be due to the context information we have managed to use, while the other deep learning baselines are matching metrics for single turns only. For a more fair comparison, we adapt the original baselines into our proposed contextual reformulation framework to incorporate context information.

In Table 5, we can see with the usage of contextual reformulation, the performances for DeepMatch, ARC, and LSTM-RNN all get boosted, which greatly indicates the effectiveness of our proposed contextual query reformulation for conversation systems. It is a useful way to incorporate context information for conversational scenarios. Our proposed DL2R has obvious improvement against the modified baseline systems. The most probable credits come from the retrieval formulation: we frame the virtual document as a posting and reply, and we integrate multi-dimensions of ranking evidences to facilitate a better ranking list. The contextual DeepMatch method very slightly outperforms DL2R on MRR evaluation. As mentioned before, MRR is useful when trying the find the best response. Since conversations are open with more than one appropriate responses, MAP and nDCG scores indicate the full capacity of the retrieval systems.

Till now, we have validated that deep learning structures, contextual reformulations and integrations of multi-dimensions of ranking evidences are effective. Next we will come to a closer look at these strategies and components for further analysis and discussions.

6.4 Analysis and Discussions

We examine the relative contributions of different strategies and individual components of our proposed model. Other than the proposed deep neural network-based learning framework, we have two other contributions: 1) contextual query reformulation to utilize context information, and 2) integration of multi-dimension of ranking evidences. We now analyze the strategies and components.

6.4.1 Reformulation Strategies

As mentioned in Section 4, we have different ways to use contextual information via contextual query reformulation strategies: *No Context*, *Whole Context*, *Add-One*, *Drop-Out* and *Combined*. The *Whole Context* strategy incorporates context information in a coarse granularity, while the *Add-One* and *Drop-Out* strategies are in a finer-granularity. We illustrate the different performance with different strategies using the DL2R framework in Table 6.

We can see that the improvement from *No Context* to *Whole Context* is rather obvious, indicating that context information is quite beneficial to find better responses under the conversational scenarios. We also have an interesting observation that *Add-One* and *Drop-Out* strategies are better than the *Whole Context* strategy. *Whole Context* strategy is a coarse-grained method which uses context information without distinguishing irrelevant contexts from relevant ones. The results indicate a proper way to use context information is important. *Drop-Out* strategy is slightly better than the *Add-One* strategy, which confirms the exemption of irrelevant context information is necessary, and in most cases, there are more relevant context sentences than irrelevant ones. The combination of all strategies performs best since it balances both the coarse-grained and fine-grained context modeling. We combine the best approximations for all possible utilizations of contexts, avoiding explosive number of combinations.

6.4.2 Components Analysis

Since we have three major components from the multi-dimensions of ranking evidences, i.e., *Query-Reply*, *Query-Posting* and *Query-Context*, we examine the contributions of such components. Note that we can retrieve responses from the virtual document consisting of a *posting-reply* pair, from the *reply* side and/or the *posting* side. But we cannot directly retrieve the response using *Query-Context*. It is pointless to run solely on the *Query-Context* part.

The first group is to run only based on *Query-Reply* and *Query-Posting*. Neither of the two components incorporates *Query-Context* information. Without context information, the proposed framework might handle single-turn conversation well enough, while generally multi-turn conversation is beyond the capability of the system. With the incorporation of the contextual information, the performance of both components get boosted. It is not surprising that the combination of all three components yield to the best results: each component characterizes the appropriateness of the response from a different aspect, and all aspects should be integrated for scoring.

7. CONCLUSIONS

In this paper, we propose to establish an automatic conversation system between humans and computers. Given a human-issued message as the query, our proposed system will return the corresponding responses based on a *deep learning-to-respond* schema. There are 3 major contributions in this work: 1) we propose a contextual query reformulation framework with ranking fusions for the conversation task; 2) we integrate multi-dimension of ranking evidences, i.e., *queries*, *postings*, *replies* and *contexts*; 3) we establish the deep neural network architecture featured with above strategies and components. We launch the conversation system based on a massive repository ($\sim$10 million *posting-reply* pairs) and run experiments to validate the proposed paradigm.

We examine the effect of our proposed DL2R model with several baselines on a series of evaluation metrics. Our method consistent-

ly and significantly outperforms the alternative baselines in terms of p@1, MAP, nDCG, and MRR. Furthermore, we have investigated further experiments for component contributions and strategy analysis. In general, context information is demonstrated to be useful for conversations, especially multi-turn conversations and all dimensions of ranking evidences are helpful. This work opens to several interesting directions for future work with regard to automatic conversation between humans and computers. For instance, we can incorporate more additional features and more conversation-oriented formulations, such as dialogue acts, conversational logics, and discourse structures, etc.

8. ACKNOWLEDGMENTS

This work is supported by the National Basic Research Program of China (No. 2014CB340505). We thank all the reviewers for their valuable comments, and thank the support from the Deep Learning Platform in Baidu Natural Language Processing Department.

9. REFERENCES

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

[2] F. Bessho, T. Harada, and Y. Kuniyoshi. Dialog system using real-time crowdsourcing and Twitter large-scale corpus. In *SIGDIAL*, pages 227–231, 2012.

[3] G. Cong, L. Wang, C.-Y. Lin, Y.-I. Song, and Y. Sun. Finding question-answer pairs from online forums. In *SIGIR*, pages 467–474.

[4] A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *Proc. Acoustics, Speech and Signal Processing*, pages 6645–6649, 2013.

[5] H. He, K. Gimpel, and J. Lin. Multi-perspective sentence similarity modeling with convolutional neural networks. In *EMNLP*, pages 1576–1586, 2015.

[6] R. Higashinaka, K. Imamura, T. Meguro, C. Miyazaki, N. Kobayashi, H. Sugiyama, T. Hirano, T. Makino, and Y. Matsuo. Towards an open domain conversational system fully based on natural language processing. In *COLING*, 2014.

[7] B. Hu, Z. Lu, H. Li, and Q. Chen. Convolutional neural network architectures for matching natural language sentences. In *NIPS*, pages 2042–2050, 2014.

[8] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.

[9] Z. Ji, Z. Lu, and H. Li. An information retrieval approach to short text conversation. *CoRR*, abs/1408.6988, 2014.

[10] N. Kalchbrenner, E. Grefenstette, and P. Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

[11] C.-J. Lee, Q. Ai, W. B. Croft, and D. Sheldon. An optimization framework for merging multiple result lists. In *CIKM '15*, pages 303–312, 2015.

[12] A. Leuski, R. Patel, D. Traum, and B. Kennedy. Building effective question answering characters. In *SIGDIAL*, pages 18–27, 2009.

[13] A. Leuski and D. Traum. NPCEditor: Creating virtual human dialogue using information retrieval techniques. *AI Magazine*, 32(2):42–56, 2011.

[14] H. Li and J. Xu. Semantic matching in search. *Foundations and Trends in Information Retrieval*, 8:89, 2014.

[15] J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*, 2015.

[16] X. Li, L. Mou, R. Yan, and M. Zhang. Stalematebreaker: A proactive content-introducing approach to automatic human-computer conversation. In *IJCAI*, 2016.

[17] Z. Lu and H. Li. A deep architecture for matching short texts. In *NIPS*, pages 1367–1375, 2013.

[18] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[19] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv:1301.3781*, 2013.

[20] L. Mou, G. Li, L. Zhang, T. Wang, and Z. Jin. Convolutional neural networks over tree structures for programming language processing. In *AAAI*, pages 1287–1292, 2016.

[21] L. Mou, H. Peng, G. Li, Y. Xu, L. Zhang, and Z. Jin. Discriminative neural sentence modeling by tree-based convolution. In *EMNLP*, pages 2315–2325, 2015.

[22] L. Mou, M. Rui, G. Li, Y. Xu, L. Zhang, R. Yan, and Z. Jin. Recognizing entailment and contradiction by tree-based convolution. *arXiv preprint arXiv:1512.08422*, 2015.

[23] M. Nakano, N. Miyazaki, N. Yasuda, A. Sugiyama, J.-i. Hirasawa, K. Dohsaka, and K. Aikawa. WIT: A toolkit for building robust and real-time spoken dialogue systems. In *SIGDIAL*, pages 150–159.

[24] E. Nouri, R. Artstein, A. Leuski, and D. R. Traum. Augmenting conversational characters with generated question-answer pairs. In *AAAI Fall Symposium: Question Generation*, 2011.

[25] H. Palangi, L. Deng, Y. Shen, J. Gao, X. He, J. Chen, X. Song, and R. Ward. Deep sentence embedding using the long short term memory network: Analysis and application to information retrieval. *arXiv preprint arXiv:1502.06922*, 2015.

[26] A. Ritter, C. Cherry, and W. B. Dolan. Data-driven response generation in social media. In *EMNLP*, pages 583–593, 2011.

[27] T. Rocktäschel, E. Grefenstette, K. M. Hermann, T. Kočiskỳ, and P. Blunsom. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*, 2015.

[28] A. Severyn and A. Moschitti. Learning to rank short text pairs with convolutional deep neural networks. In *SIGIR '15*, pages 373–382.

[29] L. Shang, Z. Lu, and H. Li. Neural responding machine for short-text conversation. In *ACL-IJCNLP*, pages 1577–1586, 2015.

[30] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *EMNLP*, pages 151–161, 2011.

[31] H. Sugiyama, T. Meguro, R. Higashinaka, and Y. Minami. Open-domain utterance generation for conversational dialogue systems using Web-scale dependency structures. In *SIGDIAL*, pages 334–338, 2013.

[32] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

[33] M. A. Walker, R. Passonneau, and J. E. Boland. Quantitative and qualitative evaluation of darpa communicator spoken dialogue systems. In *ACL*, pages 515–522, 2001.

[34] R. S. Wallace. *The Anatomy of ALICE*. Springer, 2009.

[35] H. Wang, Z. Lu, H. Li, and E. Chen. A dataset for research on short-text conversations. In *EMNLP*, pages 935–945, 2013.

[36] J. Williams, A. Raux, D. Ramachandran, and A. Black. The dialog state tracking challenge. In *SIGDIAL*, pages 404–413, 2013.

[37] Y. Xu, R. Jia, L. Mou, G. Li, Y. Chen, Y. Lu, and Z. Jin. Improved relation classification by deep recurrent neural networks with data augmentation. *arXiv preprint arXiv:1601.03651*, 2016.

[38] Y. Xu, L. Mou, G. Li, Y. Chen, H. Peng, and Z. Jin. Classifying relations via long short term memory networks along shortest dependency paths. In *EMNLP*, 2015.

[39] R. Yan. i, poet: Automatic poetry composition through recurrent neural networks with iterative polishing schema. In *IJCAI*, 2016.

[40] R. Yan, M. Lapata, and X. Li. Tweet recommendation with graph co-ranking. In *ACL*, pages 516–525, 2012.

[41] R. Yan, C.-T. Li, H.-P. Hsieh, P. Hu, X. Hu, and T. He. Socialized language model smoothing via bi-directional influence propagation on social networks. In *WWW '16*, pages 1395–1405, 2016.

[42] R. Yan, X. Wan, J. Otterbacher, L. Kong, X. Li, and Y. Zhang. Evolutionary timeline summarization: A balanced optimization framework via iterative substitution. In *SIGIR '11*, pages 745–754, 2011.

[43] R. Yan, I. E. Yen, C.-T. Li, S. Zhao, and X. Hu. Tackling the achilles heel of social networks: Influence propagation based language model smoothing. In *WWW '15*, pages 1318–1328, 2015.

[44] K. Zhai and D. J. Williams. Discovering latent structure in task-oriented dialogues. In *ACL*, pages 36–46, 2014.

[45] B. Zhang, J. Su, D. Xiong, Y. Lu, H. Duan, and J. Yao. Shallow convolutional neural network for implicit discourse relation recognition. In *EMNLP*, pages 2230–2235, 2015.