An Information Retrieval Approach to Short Text Conversation[☆]

Zongcheng Ji[a,*], Zhengdong Lu[b], Hang Li[b]

[a]*Noah's Ark Lab, Huawei Technologies, Beijing, China*
[b]*Noah's Ark Lab, Huawei Technologies, Hong Kong, China*

Abstract

Human computer conversation is regarded as one of the most difficult problems in artificial intelligence. In this paper, we address one of its key sub-problems, referred to as short text conversation, in which given a message from human, the computer returns a reasonable response to the message. We leverage the vast amount of short conversation data available on social media to study the issue. We propose formalizing short text conversation as a search problem at the first step, and employing state-of-the-art information retrieval (IR) techniques to carry out the task. We investigate the significance as well as the limitation of the IR approach. Our experiments demonstrate that the retrieval-based model can make the system behave rather "intelligently", when combined with a huge repository of conversation data from social media.

Keywords: Short Text Conversation, Information Retrieval, Learning to Rank, Learning to Match

1. Introduction

Human computer conversation is one of the most challenging AI problems, which involves language understanding, reasoning, and use of common sense knowledge. Despite a significant amount of effort on the research in the past decades, the progress on the problem is unfortunately quite limited. One of the major reasons for that is lack of large volumes of real conversation data (Chen et al., 2011; Nouri et al., 2011).

In this paper, we consider a much simplified version of the problem: one round of conversation formed by two short texts, with the former being a message from human and the latter being a response to the message from the computer. We refer to it as **short text conversation (STC)**. Thanks to the extremely large amount of short text conversation data available on social media such as Twitter[1] and Weibo[2], we anticipate that significant progress could be made in the research on the problem with the use of the big data, much like what has happened in machine translation, community question answering, etc.

Modeling a short text conversation is much simpler than modeling a complete dialogue, which often requires several rounds of interactions (e.g., a dialogue system as in Litman et al. (2000)). However, it can shed important light on understanding of the complicated mechanism of natural language dialogues and can significantly enhance the research toward the ultimate goal of passing the Turing test. The research on the problem will instantly help applications such as chatbot at a web site, automatic short-message reply on mobile phone, and voice assistant like Siri[3]. With the emergence of social media, as well as the spread of mobile devices, conversation via short texts has become an important way of communication for people in our time.

[☆]This paper is an extended version of Wang et al. (2013) with the following new content: (1) empirical verification of effectiveness of several new matching models including translation model and deep matching model, and (2) proposal of topic-word model.

[*]Corresponding author.

Email addresses: `jizongcheng@gmail.com` (Zongcheng Ji), `lu.zhengdong@huawei.com` (Zhengdong Lu), `hangli.hl@huawei.com` (Hang Li)

[1]http://twitter.com/

[2]http://weibo.com/

[3]http://en.wikipedia.org/wiki/Siri

One simple approach to STC, and perhaps the first approach which one would want to try, is to take it as an information retrieval (IR) problem, maintain a large repository of short text conversation data, and develop a conversation system mainly based on IR technologies. Given a message, the system retrieves related responses from the repository and returns the most reasonable response. That is to say, we would not generate a new response, but select the most suitable response (originally made to other messages) as reply to the current message. We refer to the former approach as **generation-based STC** and the latter approach as **retrieval-based STC**. With advanced IR technologies and a dataset with previously unthinkable volume, we would expect that the conversation system can behave almost like a human in each round of conversation.

Retrieval-based STC is similar to some IR tasks such as community question answering (CQA). In the former task, each instance consists of a message-response pair (or a post-comment pair in social media), while in the latter task, each instance consists of a question-answer pair. The differences are also evident, however. The messages tend to be longer than the responses in STC, while the answers tend to be longer than the questions in CQA. More importantly, the relations between texts are different in the two problems. The answers in CQA must be solutions to the questions, which mostly involves knowledge, while the responses in STC need only be explanations, opinions, or criticisms on the messages, which is more about appropriateness or human-likeness.

In this paper, we try to answer the question of to what extent the IR approach can effectively manipulate STC. We first propose a framework, which employs a learning to rank method for training of the ranking model and matching models as features of the ranking model. The matching models include (1) basic matching models based on cosine similarities, (2) translation model, (3) latent space model (linear model), (4) deep matching model (non-linear), and (5) topic-word model. The latent space model and deep matching model are techniques recently developed for search and question answering, and the topic-word model is devised for STC in this paper. We then conduct large scale experiments with a dataset from Weibo, which we have created and released for research on STC. Our experiments show that the retrieval-based approach can achieve fairly good performance with the precision at position one being 0.64. Experiments also show that all the matching models can significantly improve the performance. We also conduct case study to show the significance and limitation of the IR approach.

The contributions of this paper, which is an extension of our previous paper (Wang et al., 2013), include (1) proposal of the IR approach to STC, (2) empirical verification of effectiveness and restriction of the approach, (3) proposal of topic-word model for STC, and (4) development of public dataset for the research.

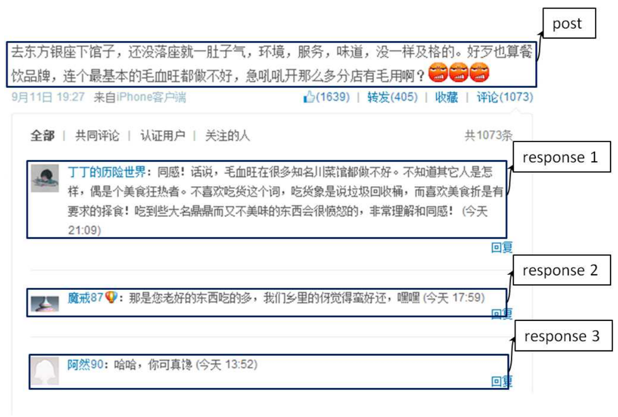
The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 shows an example of conversation on Weibo. Section 4 gives the definition of retrieval-based STC and a three-stage retrieval based framework to perform STC. Section 5 and section 6 give details on the dataset and the matching features used in the framework. Section 7 and 8 describe the experimental results and case studies, respectively. Finally, the work is concluded and future research directions are identified in Section 9.

2. Related Work

2.1. Short Text Conversation

Early work on modeling dialogues is either rule-based (Weizenbaum, 1966) or learning-based (Litman et al., 2000; Schatzmann et al., 2006; Williams and Young, 2007). These approaches require no data (e.g., rule based) or little data (e.g., reinforcement learning based) for training, but much manual effort in building the model, which is usually very costly. Furthermore, the coverage of the systems is also not satisfactory.

An alternative approach is to build a dialogue system with a knowledge base consisting of large number of question-answer pairs. For example, the system in Leuski et al. (2006) and Leuski and Traum (2011) selects the most suitable response to the current message from the question-answer pairs using a statistical language model in cross-lingual information retrieval. The major bottleneck of this approach is creation of the knowledge base (i.e., question-answer pairs). Instead of building knowledge base by hand, Chen et al. (2011) and Nouri et al. (2011) propose augmenting a knowledge base with question-answer pairs derived from texts using a question generation tool. The results show that the augmented system can answer questions about new topics, with certain performance drop on questions about existing topics. The number of question-answer pairs obtained in this way is still small (a few thousands). Furthermore, the statistical language model can only match questions and answers at the word level, not at the semantic level, which may hinder the performance of the system.



去东方银座下馆子，还没落座就一肚子气，环境，服务，味道，没一样及格的。好歹也算餐饮品牌，连个最基本的毛血旺都做不好，急吼吼开那么多分店有毛用啊？

9月11日 19:27 来自iPhone客户端 (1639) | 转发(405) | 收藏 | 评论(1073)

丁丁的历险世界：同感！话说，毛血旺在很多知名川菜馆都做不好。不知道其它人是怎样，偶是个美食狂热者。不喜欢吃货这个词，吃货象是说垃圾回收桶，而喜欢美食折是有要求的择食！吃到些大名鼎鼎而又不美味的东西会很愤怒的，非常理解和同感！(今天 21:09)

魔戒87：那是您老好的东西吃的多，我们乡里的伢觉得蛮好还，嘿嘿 (今天 17:59)

阿然90：哈哈，你可真馋 (今天 13:52)

Figure 1: An example of post and associated comments at Weibo.

All the above systems work at small scale in the sense that they can only respond to a small variety of messages. Recently, with the fast development of social media, such as community question answering and microblog services, a very large amount of conversation data becomes available. Ritter et al. (2011) investigate 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. The results show that phrase-based SMT (Koehn et al., 2007) works better than vector space model (VSM) (Salton et al., 1975) in IR in terms of BLEU score (Papineni et al., 2002). 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 this paper, we conduct short text conversation (STC) by leveraging the state-of-the-art IR technologies and the vast amount of conversation data on social media. A related but slightly different problem has been studied in Jafarpour et al. (2010), as an initial step for building a chatbot, referred to as learning to chat (L2C). L2C attempts to perform human computer conversation by utilizing machine learning and large scale dialogue data (each instance consists of several rounds of conversation). Their method first filters and then ranks responses to find the best candidate, on the basis of all information in the context. It is, thus, not directly comparable with our method in this paper.

2.2. Search

Learning to rank and semantic matching are considered state-of-the-art techniques for search (Liu, 2009; Li, 2011a,b; Li and Xu, 2014). Given a query, documents containing the query terms are first retrieved from the index. Matching between the query and each of the documents is then carried out using different models such as traditional IR model of BM25 (Robertson et al., 1995), translation model (Berger and Lafferty, 1999), and latent space model (Wu et al., 2013). The matching scores of each document are taken as features of the document. Next, the documents are assigned scores by the ranking model on the basis of their features. Finally, the documents are sorted by their ranking scores. The ranking model is trained in advance using learning to rank, and the matching models are trained in advance using semantic matching techniques (or in general learning to match). In this paper, we employ the IR techniques for short text conversation.

3. Conversation on Social Media

Weibo is a microblog service in China, similar to Twitter, on which a user can publish a short message (referred to as *post* in the remainder of the paper) visible to the public or a group of users following her/him. Just like Twitter, Weibo also has the length limit of 140 Chinese characters on each post. Users can attach a short message to a published post, with the same length limit, referred to as *comment* in this paper. Figure 1 shows an example of post and associated comments (in Chinese).

We argue that the post-comment pairs on Weibo provide a rather valuable resource for studying short text conversation between users. The comments to a post can be of flexible forms and diverse topics, as illustrated in the example

Table 1: An example of Weibo post and associated comments. The original texts are in Chinese, and we translate them into English. We do the same thing for all the examples in this paper.

Post	
User **A**:	在夏威夷的第一天，端着一大杯葡萄酒，在阳台看日落。 The first day at Hawaii. Watching sunset at the balcony with a big glass of wine in hand.
Comments	
User **B**:	好好享受，别忘了分享照片！ Enjoy it & don't forget to share your photos!
User **C**:	下次带我一起去！ Please take me with you next time!
User **D**:	打算在那呆多久？ How long are you going to stay there?
User **E**:	你什么时候作报告？ When will be your talk?
User **F**:	哈哈，此时我在做着同样的事。你住在哪个酒店？ Haha, I am doing the same thing right now. Which hotel are you staying in?
User **G**:	别炫了，老兄！此时我们还在实验室疯狂的编码呢。。 Stop showing-off, buddy. We are still coding crazily right now in the lab.
User **H**:	你真幸运！我们去火奴鲁鲁的航班延误了，我被困在了机场。现在在麦当劳嚼着薯条。 Lucky you! Our flight to Honolulu is delayed and I am stuck in the airport. Chewing French fries in MacDonald's right now.

in Table 1. With a post being a report about the user's current status (travelling to Hawaii), the comments can be a question about the user's future status, a request to the user, a greeting to the user, and so on, but are apparently all appropriate.

In many cases, the post-comment pair is self-contained, which means that one does not need any background knowledge and context information to understand the conversation (Examples of that include the comments of users **B**, **D**, **G** and **H**). In some cases, one may need extra knowledge and information to understand the conversation. For example, the comment of user **E** will be fairly elusive if taken out of the context that **A**'s Hawaii trip is for an international conference and he is going to give a talk there. We argue that the number of self-contained post-comment pairs is vast, and therefore the collected post-comment pairs can serve as a rich resource for exploring rather sophisticated patterns and structures of natural language conversations.

4. Retrieval-based Short Text Conversation

4.1. Problem Definition

Short text conversation (STC) is defined as one round of conversation via two short texts, with the former being a message from human and the latter being a response to the message given by the computer. As the first step, we formalize STC as an information retrieval (IR) problem, i.e., conduct **retrieval-based STC**. Given a message (query), the system retrieves related responses from the large repository of conversation data and returns the most reasonable response. With advanced IR technologies and a dataset with previously unthinkable volume, we would expect the conversation system can behave almost like a human in each round of conversation.

Formally, for a given query q, we select from the repository of post-comment pairs (p, r) the response r with the highest ranking score.

$$r^* = \arg\max_{(p,r)} score(q, (p, r)) \tag{1}$$

where the score is an ensemble of individual matching features.

$$score(q, (p, r)) = \sum_{i \in \Omega} \omega_i \Phi_i(q, (p, r)) \tag{2}$$

where $\Phi_i(q, (p, r)$ is the score of the i-th matching feature and ω_i is the corresponding feature weight.

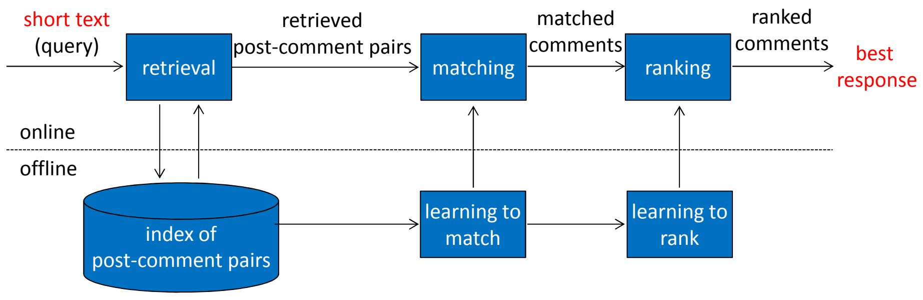


Figure 2: System architecture of retrieval-based short text conversation.

4.2. System Architecture

The system performs retrieval-based short text conversation in three-stages, as illustrated in Figure 2:

- **Stage I (retrieval)**, the system employs three fast basic linear matching models (see Section 6.1) to retrieve a number of candidate post-comment pairs for the given query q, forming a reduced candidate set $C_q^{(reduced)}$.
- **Stage II (matching)**, the system utilizes more matching models (see Section 6.2 ∼ 6.5) to further evaluate all the comments in $C_q^{(reduced)}$, returning a matching feature set $\{\Phi_i(q, (p, r), i \in \Omega\}$ for each candidate post-comment pair. The matching models are learned offline with techniques referred to as learning to match (cf., Section 6 for details).
- **Stage III (ranking)**, the system uses a linear ranking function defined in Equation (2) with the matching models as features to further evaluate all the comments (responses) in $C_q^{(reduced)}$, and assigns a ranking score to each candidate comment. Then, the system ranks the candidate comments based on their scores and selects the comment with the highest score to respond. The linear ranking function is learned offline with learning to rank techniques.

4.3. Learning of Ranking Model

We employ linear RankingSVM (Herbrich et al., 1999), a state-of-the-art method of learning to rank, to train the ranking model. We use as training data labeled post-comment pairs, as explained in Section 5.2. From the labeled data, we derive pairwise preference data $(q, (p, r)^+, (p, r)^-)$ such that $score(q, (p, r)^+) > score(q, (p, r)^-)$. Specifically, $(p, r)^+$ are selected from the labeled positive instances with respect to q, while $(p, r)^-$ are selected from the labeled negative instances. We have confirmed that the use of labeled negative instances, instead of randomly selected instances, can yield slightly better results.[4]

5. Short Text Conversation Dataset

This section introduces the dataset used for retrieval-based STC.[5] There are (1) original post-comment pairs used as the retrieval repository, and (2) labeled post-comment pairs used for training and testing different retrieval models. We give the detail of creation in the following subsections.

5.1. Original Post-Comment Pairs

The original post-comment pairs are sampled from Weibo posts published by users in a loosely connected community and the comments they received (may not be from this community).

The creation process of the dataset, as illustrated in Figure 3, consists of three consecutive steps: (1) crawling the community of users, (2) crawling their posts and the associated comments, and (3) cleaning the data, with more details described below.

[4]This is because the negative instances are collected from the top ranked candidates with several simple retrieval models, and thus they are more indicative of the difference between positive and negative instances.

[5]We have got permission from Weibo to release the dataset for research purpose and it is available at http://data.noahlab.com.hk/conversation/.

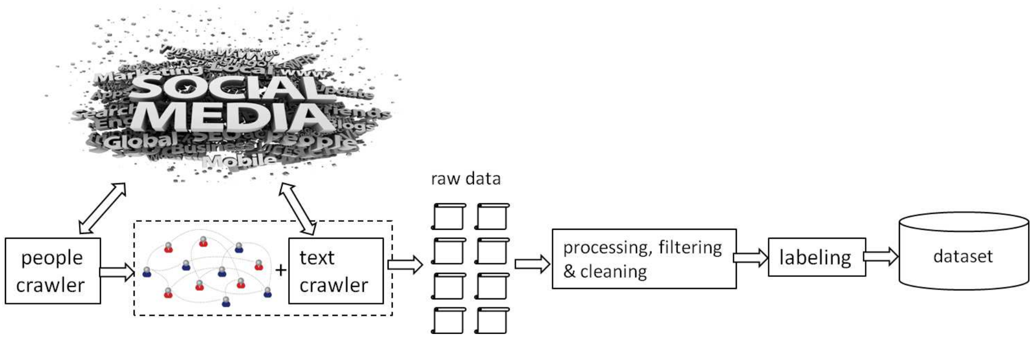


Figure 3: Diagram of the process for creating the original and the labeled post-comment pairs.

5.1.1. Sampling Strategy

We take the following sampling strategy for collecting the post-comment pairs to make the topic relatively focused. We first locate 3,200 users from a loosely connected community of Natural Language Processing (NLP) and Machine Learning (ML) in China. The community is mainly composed of professors, researchers, and students. This is done through crawling followees[6] of ten manually selected seed users who are NLP researchers active on Weibo (with no less than 2 posts per day on average) and popular enough (with no less than 100 followers).

We crawl the posts and the associated comments (not necessarily from the crawled community) for two months (from April 5th, 2013 to June 5th, 2013). The topics are relatively limited due to our choice of the users, with the most saliently ones being:

- **Research**: discussion on research ideas, papers, books, tutorials, conferences, and researchers in NLP and ML, etc;
- **General Arts and Science**: mathematics, physics, biology, music, painting, etc;
- **IT Technology**: mobile phones, IT companies, jobs opportunities, etc;
- **Life**: traveling (both touring or conference trips), food, photography, etc.

5.1.2. Processing, Filtering, and Data Cleaning

On the crawled posts and comments, we first perform a four-step filtering on the posts and comments:

- We first remove a post-comment pair if the length of the post is less than 10 Chinese characters or the length of the comment is less than 5 Chinese characters. The reason for that is two-fold: (1) if the text is too short, it can barely contain information that can be reliably captured, e.g. the following example

Post	三个了，还差两个。 Three down, two to go.

and (2) some of the posts or comments are too general to be interesting for other cases, e.g. the comment in the example below

Post	餐厅不错，强烈推荐。除了排队等待，这里什么都好。 Nice restaurant. I’d strongly recommend it. Everything here is good except the long waiting line.
Comment	哇！ Wow.

[6]When user A follows user B, A is called B’s follower, and B is called A’s followee.

Table 2: Statistics of the original and the labeled post-comment pairs.

Original Post-Comment Pairs (Used as retrieval repository)	#posts	38,016
	#comments	618,104
	#original pairs	618,104
Labeled Post-Comment Pairs (Used for training/testing retrieval models)	#posts	422
	#comments	12,402
	#labeled pairs	12,402

- In the remained post-comment pairs, we only keep the first 100 comments for each post, since we observe that after the first 100 comments there will be a non-negligible proportion of comments addressing the earlier comments rather than the original post.
- We then remove the comments that explicitly address other comments from the remained post-comment pairs.
- The last step is to filter out potential advertisements. We find out long comments that have been posted more than twice on different posts are likely to be advertisements and we scrub them out of the remained post-comment pairs.

Then, for the remained posts and comments, we remove punctuation marks and emotions, and use ICTCLAS (Zhang et al., 2003) for Chinese word segmentation.

Finally, we get 38,016 Weibo posts and their corresponding 618,104 comments, forming 618,104 *original* post-comment pairs, which are used as the retrieval repository in all the experiments. The statistics of the dataset are summarized in Table 2.

5.2. *Labeled Post-Comment Pairs*

This subsection introduces the creation process of the labeled post-comment pairs, which are used for training and testing different retrieval models for STC.

We employ a pooling strategy widely used in information retrieval for getting the instances to label (Voorhees, 2002). More specifically, for a given query, we use each of three basic retrieval models to select top 10 comments (see Section 6.1 for the description of the basic matching models), and merge them to form a much reduced candidate set with size ≤ 30. Then we assign the comments in the reduced candidate set into "suitable" and "unsuitable" categories. Basically we consider a comment suitable for a given query if we cannot tell whether it is an original comment. More specifically the suitability of a comment is judged based on the following three criteria[7]:

- **Semantic Relevance**: This requires the content of the comment to be semantically relevant to the post. As shown in the example right below, the query is about soccer, and so is comment 1, and hence it is semantically relevant, whereas comment 2 is about food and hence semantically irrelevant.

Query	英格兰禁区里老是八个人...... There are always 8 English players in their own penalty area. Unbelievable!
Comment 1	哈哈哈仍然是0：0。还没看到进球。 Haha, it is still 0:0, no goal so far.
Comment 2	英格兰的食物真可怕! The food in England is horrible.

Another important aspect of semantic relevance is entity association. This requires that the entities in the comment to be strongly associated with those in the query. In other words, if the query is about entity A, while

[7]Note that although our criteria in general favor short and general comments like "说得很好！(Well said!)" or "太好了！(Nice!)", most of the general comments have already been filtered out due to their lengths (see Section 5.1.2).

the comment is about entity B, they are very likely to be mismatched. As shown in the following example, where the query is about Paris, while comment 2 talks about London:

Query	巴黎的天空！暖暖的太阳+ 阵阵的微风，真让人不舍得离开。 The sky in Paris! Warm sun + bursts of breeze. Really hate to leave.
Comment 1	好好享受你在巴黎的时间吧。 Enjoy your time in Paris.
Comment 2	我希望我现在在伦敦。 Man, I wish I am in London right now.

This is however not absolute, since a comment containing a different entity could still be sound, as demonstrated by the following two comments to the query above:

Comment 1	好好享受你在法国的时间吧。 Enjoy your time in France.
Comment 2	伦敦的秋天也很漂亮。 The fall of London is nice too.

- **Logic Consistency**: This requires the content of the comment to be logically consistent with the query. For example, in the table right below, the query states that the Huawei mobile phone "Honor" is already in the market of mainland China. Comment 1 talks about a personal preference over the same phone model (hence logically consistent), whereas comment 2 asks a question to which the answer is already clear from the query (hence logically inconsistent).

Query	华为荣耀手机在中国大陆大卖！ HUAWEI's mobile phone, Honor, sells well in mainland China.
Comment 1	华为荣耀手机是我最喜欢的一款手机。 HUAWEI Honor is my favorite phone
Comment 2	华为荣耀手机什么时候会在中国大陆上市？ When will HUAWEI Honor get to the market in mainland China?

- **Speech Act Agreement**: Another important factor in determining the suitability of a comment is speech act. For example, when a question is posed in the Weibo post, a certain act (e.g., answering or forwarding it) is expected. In the example below, the query asks a special question about location. Comment 1 forwards and comments 2 answers the question, whereas comment 3 is an imperative sentence and therefore does not correspond well in speech act.

Query	有人知道后年的KDD 会在哪里举行呢？ Any one knows where KDD will be held the year after next?
Comment 1	同问。希望在欧洲 co-ask. Hopefully Europe
Comment 2	听说在纽约 New York, as I heard
Comment 3	请帮忙扩散这条关于KDD 的信息。 Please help me distribute the information of KDD.

Finally, we manually label 422 queries and their associated comments, about 30 comments for each post. Note that (1) the labeling is only on a small subset of the 38,016 posts, and (2) for each selected (query) post, the labeled comments are *not originally* given to it. The statistics of this part of dataset are also summarized in Table 2.

6. Matching Features

In this section, we introduce matching features (calculated from matching models) used in our retrieval-based STC. Three basic linear matching models are first introduced as the baselines. Then, a translation-based language model (TransLM) is introduced for alleviating the lexical gap problem. Next, a deep matching model (DeepMatch) is described, which matches query and response (comment) with a deep architecture. After that, a topic-word model (TopicWord) is explained for addressing the topic alignment of query and response (comment). Finally, other simple matching features used in our retrieval-based STC are described.

6.1. Basic Linear Matching Models

We use the following three basic linear matching models for fast retrieval in **Stage I**. Moreover, these matching models are also used in **Stage II** to generate three matching features for each post-comment pair.

6.1.1. Query-Response Similarity

Here we use a simple vector space model for measuring the similarity between a query q and a candidate response r

$$sim_{Q2R}(\mathbf{q}, \mathbf{r}) = \frac{\mathbf{q}^{\mathsf{T}}\mathbf{r}}{||\mathbf{q}||||\mathbf{r}||} \tag{3}$$

where $\mathbf{q}$ and $\mathbf{r}$ are respectively the TF-IDF vectors of q and r.

Although it is not necessarily true that a good response has many common words as the query, but this measurement is often helpful in finding relevant responses. For example, when the query and the candidate response both have "National Palace Museum in Taipei", it is a strong signal that they are about similar topics. Unlike other semantic matching features, this simple similarity requires no learning and works on infrequent words. Our empirical results show that it can often capture the query-response relation which semantic matching features cannot.

6.1.2. Query-Post Similarity

The basic idea here is to find posts (messages) similar to the query q and use their comments (responses) as the candidates. Again we use the vector space model for measuring the query-post similarity

$$sim_{Q2P}(\mathbf{q}, \mathbf{p}) = \frac{\mathbf{q}^{\mathsf{T}}\mathbf{p}}{||\mathbf{q}||||\mathbf{p}||} \tag{4}$$

where $\mathbf{q}$ and $\mathbf{p}$ are respectively the TF-IDF vectors of q and p.

The assumption here is that if a post p is similar to the query q, its associated comments (responses) might be appropriate for q. It however often fails, especially when a response to p addresses parts of p not contained by q, which fortunately can be alleviated when combined with other measures.

6.1.3. Query-Response Matching in Latent Space

This particular matching function relies on mapping of posts and responses in the original vector spaces to a low-dimensional latent space, learned from data. The matching score between a query q and a candidate response r can be measured as the inner product between their images in the latent space.

$$LatentMatch(\mathbf{q}, \mathbf{r}) = \mathbf{q}^{\mathsf{T}} L_{\mathbf{q}} L_{\mathbf{r}}^{\mathsf{T}} \mathbf{r} \tag{5}$$

This is to capture the semantic matching between a post and a response, which may not be well captured by a word-to-word matching. We find the mapping functions $L_{\mathbf{q}}$ and $L_{\mathbf{r}}$ through a large number of query-response pairs and a large margin variant of the method in Wu et al. (2013).

$$
\begin{aligned}
\arg\min_{L_{\mathbf{q}},L_{\mathbf{r}}} \ & \sum_i \max(1 - \sum_i \mathbf{q}_i^\top L_{\mathbf{q}} L_{\mathbf{r}}^\top \mathbf{r}_i, 0) \\
\text{s.t. } & ||L_{n,\mathbf{q}}||_1 \le \mu_1, n = 1, 2, ..., N_{\mathbf{q}} \\
& ||L_{m,\mathbf{r}}||_1 \le \mu_1, m = 1, 2, ..., N_{\mathbf{r}} \\
& ||L_{n,\mathbf{q}}||_2 = \mu_2, n = 1, 2, ..., N_{\mathbf{q}} \\
& ||L_{m,\mathbf{r}}||_2 = \mu_2, m = 1, 2, ..., N_{\mathbf{r}}
\end{aligned} \tag{6}
$$

where i indices the original post-comment pairs.

Our experiments (see Table 14) indicate that this simple linear model can learn meaningful semantic matching patterns, due to its effective use of massive data. For example, the image of the word "Italy" in the post in the latent space matches well the words "Sicily", "Mediterranean sea" and "travel". Once the mapping $L_{\mathbf{q}}$ and $L_{\mathbf{r}}$ are learned, the semantic matching score $\mathbf{q}^\top L_{\mathbf{q}} L_{\mathbf{r}}^\top \mathbf{r}$ will be utilized as a feature for modeling the overall suitability of r as a response to q.

6.2. *Translation-based Language Model*

6.2.1. *Motivation*

With the three basic matching models and some simple features (see Section 6.5), the model for retrieval-based STC can achieve fairly good performance (see Section 7.2). However, there are also some cases in which the model fails. One of the serious problems is the *lexical gap*, which is the major challenge for most information retrieval tasks, between the query and the candidate post-comment pairs. Table 3 shows a real example of the lexical gap problem. Two candidate responses are suitable to the query, while their ranking is very low in the top 30 candidates. The main reason is that there is no word overlap between the candidate responses and the query, although there is one common word "晚安(Good Night)" between the original posts and the query.

Table 3: An example of the lexical gap problem between the query and the candidate post-comment pairs. The "Labels" column indicates whether the candidate responses are suitable to the query. The "Candidate Responses" and "Original Posts" columns list the candidate responses and their original posts, respectively. The "Rank" column shows the ranking of the 2 responses in the top 30 candidates with the basic matching models.

Query	小舟划向夕阳（摄于西西里岛附近）。晚安！ A boat rowed off into the sunset (taken in the vicinity of Sicily). Good Night!		
Labels	**Candidate Responses**	**Original Posts**	**Rank**
Suitable	日落夜不黑的欧洲夜晚 Sunset night, not black night in Europe	苏黎世日落。大家晚安！ Zurich sunset. Good night!	22
Suitable	种么美，想起《爱在黄昏日落时》的场景 What a beauty! I think of the "Love in the Sunset" scene	巴黎日落。大家晚安！ Paris sunset. Good night!	25

6.2.2. *Model Description*

To alleviate the lexical gap problem, we employ the state-of-the-art translation-based language model (TransLM) (Xue et al., 2008) with a small modification for retrieval-based STC. Given a query q and a candidate post-comment pair (p, r), the ranking function based on TransLM is written as

$$P_{TransLM}(q|(p,r)) = \prod_{w \in q} P_{TransLM}(w|(p,r)) \tag{7}$$

$$P_{TransLM}(w|(p,r)) = (1-\alpha)P_{mx}(w|(p,r)) + \alpha P_{ml}(w|C) \tag{8}$$

$$
\begin{aligned}
P_{mx}(w|(p,r)) = & (1-\beta)\left[(1-\gamma)P_{ml}(w|p) + \gamma \sum_{t \in p} T(w|t) P_{ml}(t|p)\right] \\
& + \beta\left[(1-\gamma)P_{ml}(w|r) + \gamma \sum_{t \in r} T(w|t) P_{ml}(t|r)\right]
\end{aligned} \tag{9}
$$

where $P_{ml}(w|p)$, $P_{ml}(w|r)$, and $P_{ml}(w|C)$ are the unigram language models (LM), which are estimated with maximum likelihood, for the post part p, the response part r and the whole collection C, respectively. $T(w|t)$ is the probability of translating a word t in p or r into a word w in q. $\sum_{t \in p} T(w|t)P_{ml}(t|p)$ and $\sum_{t \in r} T(w|t)P_{ml}(t|r)$ are the translation models (Trans) for the post part and response part, respectively. To bridge the lexical gap between queries and candidate post-comment pairs, the two translation models allow a post and its response translate any one of their words t to a different but semantically related query word w with a non-zero probability. α is the Jelinek-Mercer smoothing factor (Zhai and Lafferty, 2001). β is the interpolation parameter between the post model and the response model. γ is the parameter to balance between the unigram language model and the translation model for alleviating the lexical gap problem and the self-translation problem (Xue et al., 2008).

The main difference between our TransLM for retrieval-based STC and that for question retrieval in Community Question Answering (CQA) (Xue et al., 2008) is that we add an additional part $\sum_{t \in r} T(w|t)P_{ml}(t|r)$ in the response part while Xue et al. (2008) does not do that in the answer part. The reasons are two-fold: (1) we focus on finding suitable responses given a query while Xue et al. (2008) focus on finding similar questions given a query question, and (2) the responses tend to be shorter than the posts in STC while the answers tent to be longer than the questions in CQA.

6.2.3. *Learning Translation Probabilities*

The performance of the translation-based language model relies on the quality of the word-to-word translation probabilities. We follow the method of Xue et al. (2008) to learn the word translation probabilities. In our experiments, original post-comment pairs are used for training, and the GIZA++[8] (Och and Ney, 2003) toolkit is used to learn the translation model. There are two different settings of source and target for constructing the parallel corpus: (1) set the posts as the source and the responses as the target, i.e., collection $(p, r)_1, ..., (p, r)_n$, and (2) set the responses as the source and the posts as the target, i.e., collection $(r, p)_1, ..., (r, p)_n$. Following Xue et al. (2008), a pooling strategy is adopted to combine the above two collections of pairs to get a pooled collection $(p, r)_1, ..., (p, r)_n, (r, p)_1, ..., (r, p)_n$. Moreover, we also filter low-frequency words, whose frequency is less than 10.

6.3. *Deep Matching Model*

6.3.1. *Motivation*

The matching models above are linear models. These models, although proven to be effective, are insufficient for capturing the rich structures in matching complicated objects like texts. We propose employing a new deep neural network model, referred to it as deep matching model (DeepMatch) in the paper, to model the complicated matching relations between query and candidate response in retrieval-based STC.[9] This new architecture is mainly based on the following two intuitions:

- **Localness**: There are salient local structures in the semantic space of parallel texts to be matched, which can be roughly captured by the co-occurrence pattern of words. This localness however should not prevent two "distant" components from correlating with each other on a higher level, hence call for the hierarchical characteristic of the model;
- **Hierarchy**: The decision making for matching has different levels of abstraction. The local decisions, capturing the interactions between semantically related words, will be combined later layer-by-layer to form the final and global decision on matching.

6.3.2. *Model Description*

The deep matching model (DeepMatch) consists of two parts: (1) many bilinear local matching models, and (2) a deep neural network to further combine the local matching models to generate the final matching score. Each local matching model (indexed k) is in charge of a small subset of words in both short texts ($\mathbf{x}$ and $\mathbf{y}$) and a pair of projection matrices ($L_x^{(k)}$, $L_y^{(k)}$). The score from the k^{th} matching model is given as follows

$$a^{(k)}(\mathbf{x}, \mathbf{y}) = f^{(k)}\left((\mathbf{x}^{(k)})^\top L_x^{(k)} (L_y^{(k)})^\top \mathbf{y}^{(k)} + b^{(k)}\right), \quad k = 1, \cdots, K \tag{10}$$

[8]https://code.google.com/p/giza-pp/

[9]The detail can be found in Lu and Li (2013).

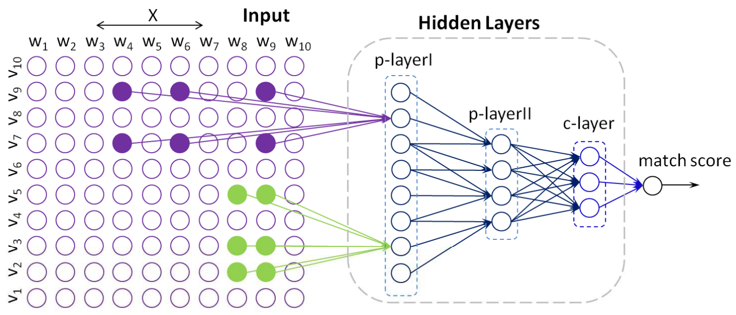

X
Input
Hidden Layers
w_1 w_2 w_3 w_4 w_5 w_6 w_7 w_8 w_9 w_{10}
v_1 v_2 v_3 v_4 v_5 v_6 v_7 v_8 v_9 v_{10}
Y
p-layerI
p-layerII
c-layer
match score

Figure 4: An illustration of the architecture of deep matching model.

where $f^{(k)}(\cdot)$ is a function with sigmoid shape for mapping the matching score to $(0, 1)$. Those local matching decisions are then used as input to feed into a multi-layer perceptron to calculate the final matching score. The overall architecture of DeepMatch is given in Figure 4.

6.3.3. *Model Training*

The training of DeepMatch is divided into two phases: (1) bilingual topic modeling for finding potentially matched subsets (topics) of word-pairs and building the model architecture, and (2) training of the parameters of the local topic models and deep neural network. In this paper, we train the deep matching model with a ranking-based objective. More specially, we employ a large margin objective defined on preference pairs in ranking. Suppose that we are given the following triples $(\mathbf{x}, \mathbf{y}^+, \mathbf{y}^-)$ from the oracle, with $\mathbf{x}$ $(\in \mathcal{X})$ matched with $\mathbf{y}^+$ better than with $\mathbf{y}^-$ (both $\in \mathcal{Y}$). The ranking-based objective is defined as follows

$$\mathcal{L}(\mathcal{W}, \mathcal{D}_{trn}) = \sum_{(\mathbf{x}_i, \mathbf{y}_i^+, \mathbf{y}_i^-) \in \mathcal{D}_{trn}} e_{\mathcal{W}}(\mathbf{x}_i, \mathbf{y}_i^+, \mathbf{y}_i^-) + R(\mathcal{W}), \tag{11}$$

where $R(\mathcal{W})$ is the regularization term and $e_{\mathcal{W}}(\mathbf{x}_i, \mathbf{y}_i^+, \mathbf{y}_i^-)$ is the error for triple $(\mathbf{x}_i, \mathbf{y}_i^+, \mathbf{y}_i^-)$, given by the following large margin form:

$$e_i = e_{\mathcal{W}}(\mathbf{x}_i, \mathbf{y}_i^+, \mathbf{y}_i^-) = \max(0, m + \mathsf{s}(\mathbf{x}_i, \mathbf{y}_i^-) - \mathsf{s}(\mathbf{x}_i, \mathbf{y}_i^+)),$$

with parameter $0 < m$ controlling the margin. In the experiments we use $m = 2$, but we find that the results are rather insensitive to the values of m varying in a fairly large range.

6.4. *Topic-Word Model*

6.4.1. *Motivation*

The matching models above are mainly for representing semantic relevance between query and response. They do not capture the matching relations between the main topics of the query and the response. Table 4 shows a real example of the problem. The top 2 candidate responses are unsuitable to the query, while the first suitable response is ranked at 9. The main reason is that the word "菜鸟(rookie)" has higher term frequencies in the query and the unsuitable responses, and thus dominates their matching scores. However, the main topics of the query is not "菜鸟(rookie)" but "代码控制工具(code control tool) SVN GIT", and thus the top 2 candidate responses are not suitable. One possible solution is to identify the topic words of the query and the candidate responses and give higher weights to the matching of the topic words. This may alleviate the topic mismatch problem and improve the performance of STC. In Table 4, the topic words of the query and the candidate responses are shown in bold. After identifying the topic words, the first two responses clearly become unsuitable to the query, while the third response having the same topic becomes suitable.

One key question is how to differentiate topic words from the other words in a short text. We propose a method for the task in the following subsections.

Table 4: An example to show that matching of topic words is necessary for retrieval-based STC. The “Labels” column indicates whether the candidate responses are suitable to the query. The “Rank” column shows the ranking of the 3 responses in the top 30 candidates with the basic matching models. The words in bold are the topic words of the post and responses.

Query	选用**代码控制工具** 的时候，还是用**SVN**，不要用**GIT**，虽然**GIT** 比**SVN** 强大很多。因为如果团队里有一个菜鸟，他各种各样关于**GIT** 的问题会烦死你。**SVN** 的优势就是极其简单，菜鸟也可以很快掌握 When choosing the **code control tool**, use **SVN** instead of **GIT**, although **GIT** is more powerful than **SVN**. The reason is that if a team has a rookie, he will bother you with all sorts of questions about **GIT**. The advantage of **SVN** is it is so simple that a rookie can grasp it quickly enough.

Labels	**Candidate Responses**	**Rank**
Unsuitable	我是**保密员** 、**菜鸟** 与**90后** **努力** 为了不成为老**菜鸟** I am a **confidential staff**, **rookie** and **the generation after 90s**, **making efforts** in order not to become an old **rookie**.	1
Unsuitable	我，还有很多用户。都是**菜鸟** 。 I and also a lot of users are **rookies**.	2
Suitable	一个**版本控制** 的工具 A **version control tool**	9

Table 5: Features for predicating the probability of word *w* being a topic word in a short text.

#	**Features**	**Description**
1	TF	Term frequency of *w* in the short text
2	IDF	Inverse document frequency of *w* in the whole collection
3	SF	Number of sentences in the short text that contain *w*
4	First	Whether *w* exists in the first sentence
5	Last	Whether *w* exists in the last sentence
6	NE	Whether *w* is a named entity (NE)
7	NE_First	Whether *w* is NE in the first sentence
8	NE_Last	Whether *w* is NE in the last sentence
9	POS	Part of speech of *w*

6.4.2. Learning Topic Words

A short text, such as a post or a comment in Weibo, usually centers around a specific theme, which is usually captured by a number of words in the text. We refer to the words as *topic words*. Examples are show in Table 4.

In this paper, we employ a probabilistic approach based on logistic regression to compute the probability of a word being the topic word of a short text. In the logistic regression model, we specifically compute $P(topic|w)$ as follows, where w denotes a word and *topic* denotes a binary variable representing whether or not w is a topic word.

$$\log\left(\frac{P(topic|w)}{1 - P(topic|w)}\right) = \vec{\omega} \cdot \vec{x} + c \tag{12}$$

$$P(topic|w) = \frac{e^{\vec{\omega}\cdot\vec{x}+c}}{1 + e^{\vec{\omega}\cdot\vec{x}+c}} \tag{13}$$

where $\vec{x}$ is a short text, such as a post or a comment in Weibo, represented as a vector of features, $\vec{\omega}$ is a vector of weights associated with the features, and c is a constant.

In the logistic regression model we make use of the features, as listed in Table 5. The first two features (TF, IDF) are from the traditional term weighting schemes in information retrieval. The third feature (SF) is based on the observation that words appearing in multiple sentences in the short text are more likely to be topic words. The next two features (First, Last) represent the positions of word in the short text. Our observation is that topic words are most likely to appear in the first or the last sentence. The next three features (NE, NE_First, NE_Last) are named entity related features. The last feature (POS) is the part of speech of a word, which is based on the fact that topic words are usually denoted by nouns and verbs.

Table 6: Statistics of topic words.

#topic words	#non-topic words	#total words
847	1,161	2,008

Table 7: Topic word distributions over word positions, parts of speech and named entities.

Topic word distribution over word positions	In the first sentence	83.33%
	In the last sentence	15.12%
	In the other sentences	1.54%
Topic word distribution over parts of speech (POS)	Noun	60.33%
	Verb	26.33%
	Adjective	8.61%
	Others	4.72%
Topic word distribution over named entities (NE)	Is NE	12.04%
	Not NE	87.96%

To create the training data, we randomly select 200 short texts (including posts and comments) to label their topic words. We make labeling judgments on all words in a short text (except some stop words), as our model is applied to words. Each word is assigned to one of the two classes, "positive" class and "negative" class, depending on whether or not it is a topic word. The judgments are made based on the principle that the topic words of a short text should be indicative to the main theme of the short text. Finally, we obtain 2,008 words for the 200 short texts. Table 6 summarizes the statistics of the dataset and Table 7 lists the topic word distributions over word positions, parts of speech and named entities in the dataset. As shown in Table 7, topic words are typically nouns in the first sentence of a short text.

We use ICTCLAS (Zhang et al., 2003) to obtain the part of speech and named entity information and LIBLINEAR (Fan et al., 2008) to build the logistic regression model and predict the probability of a word being a topic word. The accuracy of our model is 81.57%.

6.4.3. Model Description

With the trained model, we can assign a probability value to each word in the short text, which indicates the probability of the word being a topic word. The question is how to use this information in retrieval-based STC. In this work, we simply take the probability values as term weights and use them to calculate the similarities between a query q and a candidate response r or its original post p with a vector space model, which are similar to the query-response similarity and query-post similarity introduced in Section 6.1.

$$sim_{Q2R\_TopicWord}(\mathbf{q}, \mathbf{r}) = \frac{\mathbf{q}^{\mathrm{T}}\mathbf{r}}{||\mathbf{q}||||\mathbf{r}||} \tag{14}$$

$$sim_{Q2P\_TopicWord}(\mathbf{q}, \mathbf{p}) = \frac{\mathbf{q}^{\mathrm{T}}\mathbf{p}}{||\mathbf{q}||||\mathbf{p}||} \tag{15}$$

where $\mathbf{q}$, $\mathbf{p}$ and $\mathbf{r}$ are respectively the vectors of q, p and r with topic word probability $P(topic|w)$ as weight.

6.5. Other Simple Matching Features

We also use some other simple matching features as follows:

- Longest Common String (LCS): this feature measures the length of the longest common string between the query q and the candidate response r, which is useful for capturing the quotes common in microblog comments and is also fairly robust to errors in Chinese word segmentation.

$$LCS_{Q2R}(q, r) = |LCS(q, r)| \tag{16}$$

- Co-occurrence features: these features represent the size, the rate, the sum, and the average of IDF values of the co-occurring words between the query q and the candidate response r or its original post p.

$$cooccur\_size(x, y) = |cooccur(x, y)| \tag{17}$$

$$cooccur\_rate(x, y) = \frac{|cooccur(x, y)|}{|y|} \tag{18}$$

$$cooccur\_sumIDF(x, y) = \sum_{w \in cooccur(x,y)} IDF(w) \tag{19}$$

$$cooccur\_averageIDF(x, y) = \frac{\sum_{w \in cooccur(x,y)} IDF(w)}{|cooccur(x, y)|} \tag{20}$$

where x stands for the query q and y stands for the candidate response r or its original post p, $cooccur(x, y)$ is the set of common words between x and y.

7. Experiments

We conduct experiments and report results on retrieval-based STC with the dataset we have created.

7.1. Experimental Setup

7.1.1. Evaluation Metrics

We evaluate the performance of different retrieval models for STC based on the following two metrics: **Mean Average Precision** (MAP) and **Precision@1** (P@1). MAP rewards methods that return suitable responses on the top and also rewards methods that return correct ranking of responses. P@1 reports the fraction of suitable responses among the top 1 responses retrieved. All the results reported below are based on 5-fold cross-validation on the 422 queries. We also perform a significance test using a paired ***t***-test with a significant level of 0.05.

We measure the performance of the logistic regression classifier for learning topic words in terms of **Accuracy** based on 5-fold cross-validation on the 2008 labeled words.

7.1.2. Parameter Settings

There are five parameters to be set in our experiments. We tune the best parameters with 5-fold cross-validation. Finally, we set $\alpha = 0.8$ as the Jelinek-Mercer smoothing factor in Equation (8), $\beta = 0.9$ to interpolate the post model and the response model in Equation (9), $\gamma = 0.5$ to interpolate the unigram language model and the translation model in Equation (9). We set $c = 0$ in Equation (12). When training the weighs of features in Equation (2) with linear RankingSVM (Herbrich et al., 1999), we use a fixed penalty parameter (i.e., 50), as the performance is fairly insensitive to the choice of the parameter.

7.2. Results of Basic Linear Matching Models

We first evaluate the performance of the three basic linear matching models combined with the simple matching features in the learning to rank framework with Equation (2).

The results are shown in Table 8. In the table, Q2R stands for the features based on the query-response similarity (Section 6.1.1) and query-response related simple matching features (Section 6.5). Q2P stands for the features based on the query-post similarity (Section 6.1.2) and query-post related simple matching features (Section 6.5). LatentMatch stands for the latent matching feature introduced in Section 6.1.3. As shown in the table, combining the three basic linear matching models with all the simple matching features achieves fairly good performance (row 4) and we name this model as Baseline, which will be used in the following subsections. In particular, we find that the LatentMatch feature helps slightly improve the overall performance on P@1.

Table 8: Comparison of different choices of features, where Q2R stands for the features based on the query-response similarity and query-response related simple matching features, Q2P stands for the features based on the query-post similarity and query-post related simple matching features, and LatentMatch stands for the latent matching feature. Row 4 combines the three basic linear matching models with all the simple matching features.

#	**Model**	**MAP**	**P@1**
1	Q2R	0.565	0.489
2	Q2R+Q2P	0.621	0.567
3	Q2R+LatentMatch	0.575	0.513
4	Q2R+Q2P+LatentMatch (Baseline)	0.621	0.574

Table 9: Comparison of different combinations of the matching features for retrieval-based STC. %impr_MAP and %impr_P@1 stand for the percentage of improvements of the current model over the Baseline in terms of MAP and P@1, respectively.

#	**Model**	**MAP**	**P@1**	**%impr_MAP**	**%impr_P@1**
5	Baseline	0.621	0.574	—	—
6	Baseline+TransLM	0.641	0.605	2	3.1
7	Baseline+DeepMatch	0.628	0.587	0.7	1.3
8	Baseline+TopicWord	0.635	0.586	1.4	1.2
9	Baseline+TransLM+DeepMatch	0.643	0.625	2.2	5.1
10	Baseline+TransLM+TopicWord	0.653	0.623	3.2	4.9
11	Baseline+DeepMatch+TopicWord	0.641	0.606	2	3.2
12	Baseline+TransLM+DeepMatch+TopicWord	**0.654**	**0.637**	**3.3**	**6.3**

7.3. Results of Combining all the Features

Then, we further incorporate TransLM, DeepMatch and TopicWord as matching features into the learning to rank framework with Equation (2).

Table 9 shows the comparison of different combinations of the matching features for retrieval-based STC. Baseline stands for the model, which combines the three basic linear matching models with all the simple matching features, based on the learning to rank framework (see row 4 in Table 8). From the table, we can see that the three new matching features can significantly improve the retrieval performance. When combining all the three new features with Baseline, the model achieves the best performance, which outperforms Baseline by 3.3 percent and 6.3 percent in terms of MAP and P@1, respectively (row 12 vs. row 5).

In order to clearly see the contributions of the three new features, we make comparisons between the model with and without each of the features. Table 10, 11 and 12 show the contributions of TransLM, DeepMatch and TopicWord, respectively. Take Table 10 as example, X and X+TransLM stands for the models without and with TransLM, respectively. X includes Baseline, Baseline+DeepMatch, Baseline+TopicWord, and Baseline+DeepMatch +TopicWord. %impr means the percentage of improvements of X+TransLM over X in terms of MAP and P@1. Table 11 and 12 are similar to Table 10. From the three tables, we have the following findings:

- The feature TransLM can bring at least 1.3 and 3.1 percent improvements in terms of MAP and P@1, respectively.
- The feature DeepMatch can bring at least 1.3 percent improvements in terms of P@1, although it brings no much improvement in terms of MAP.
- The feature TopicWord can bring at least 1.1 and 1.2 percent improvements in terms of MAP and P@1, respectively.

From the above analysis, we find that all the three features make significant contributions to the enhancement of performance. In addition, TransLM contributes the most, TopicWord the second largest, and DeepMatch the least.

Table 10: Contributions of TransLM as a matching feature in the learning to rank framework for retrieval-based STC. X and X+TransLM stands for the models without and with TransLM, respectively. X includes Baseline, Baseline+DeepMatch, Baseline+TopicWord, and Baseline+DeepMatch+TopicWord. %impr means the percentage of improvements of X+TransLM over X in terms of MAP and P@1.

	Model	Baseline	Baseline+DeepMatch	Baseline+TopicWord	Baseline+DeepMatch+TopicWord
MAP	X	0.621	0.628	0.635	0.641
	X+TransLM	0.641	0.643	0.653	0.654
	%impr	2	1.5	1.8	1.3
P@1	X	0.574	0.587	0.586	0.606
	X+TransLM	0.605	0.625	0.623	0.637
	%impr	3.1	3.8	3.7	3.1

Table 11: Contributions of DeepMatch as a matching feature in the learning to rank framework for retrieval-based STC. X and X+DeepMatch stands for the models without and with DeepMatch, respectively. X includes Baseline, Baseline+TransLM, Baseline+TopicWord, and Baseline+TransLM+TopicWord. %impr means the percentage of improvements of X+DeepMatch over X in terms of MAP and P@1.

	Model	Baseline	Baseline+TransLM	Baseline+TopicWord	Baseline+TransLM+TopicWord
MAP	X	0.621	0.641	0.635	0.653
	X+DeepMatch	0.628	0.643	0.641	0.654
	%impr	0.7	0.2	0.6	0.1
P@1	X	0.574	0.605	0.586	0.623
	X+DeepMatch	0.587	0.625	0.606	0.637
	%impr	1.3	2	2	1.4

Table 12: Contributions of TopicWord as a matching feature in the learning to rank framework for retrieval-based STC. X and X+TopicWord stands for the models without and with TopicWord, respectively. X includes Baseline, Baseline+TransLM, Baseline+DeepMatch, and Baseline+TransLM+DeepMatch. %impr means the percentage of improvements of X+TopicWord over X in terms of MAP and P@1.

	Model	Baseline	Baseline+TransLM	Baseline+DeepMatch	Baseline+TransLM+DeepMatch
MAP	X	0.621	0.641	0.628	0.643
	X+TopicWord	0.635	0.653	0.641	0.654
	%impr	1.4	1.2	1.3	1.1
P@1	X	0.574	0.605	0.587	0.625
	X+TopicWord	0.586	0.623	0.606	0.637
	%impr	1.2	1.8	1.9	1.2

8. Case Study

To get a better understanding of the effectiveness of the matching features, we conduct case study of the features. We illustrate the results through several examples. Section 8.1 shows the effectiveness of the basic linear matching features. Sections 8.2, 8.3, and 8.4 show the effectiveness of using translation-based language model, deep matching model and topic-word model. Section 8.5 gives several examples which are not addressed well with our current model and we will leave them to future work.

8.1. The Effectiveness of Basic Linear Matching

The basic linear matching features are mostly vector-space based, which are fairly good at capturing semantic relevance, as illustrated in Table 13. The suitable responses are retrieved mainly because they have common words with the queries. The experiments also show that we may find interesting and suitable responses that have no common words with the query, as show in the example in Table 14.

8.2. The Effectiveness of Translation-based Language Model

The experimental results show that TransLM has superior performance when used as a feature in the learning to rank framework. Candidate post-comment pairs that do not share many common words with the query tend to be ranked low by the other matching models. However, the translation-based model is able to fill the lexical gap and find

Table 13: Two illustrative examples of the effectiveness of basic matching models.

Query 1	这是西班牙的一个拥有500人口的小城镇，你猜怎么着，他们甚至有一个赌场！ It is a small town in Spain with 500 population, and guess what, they even have a casino!
Suitable Response 1	如果你到西班牙旅游，你需要花一些时间在那里。 If you travel to Spain, you need to spend some time there.
Query 2	引自本杰明·富兰克林的一个名言：“我们都是天生的无知，但必须努力保持着愚蠢。” One quote from Benjamin Franklin: “We are all born ignorant, but one must work hard to remain stupid.”
Suitable Response 2	本杰明·富兰克林是位智者，美国的开国元勋之一。 Benjamin Franklin is a wise man, and one of the founding fathers of USA.

Table 14: An illustrative example of the effectiveness of latent space matching model.

Query	英格兰禁区里老是八个人...... Eight England players stand in the penalty area ...
Suitable Response 1	那场比赛真经典。 What a classic match.
Suitable Response 2	哈哈哈仍然是1：0。还没看到进球。 Hahaha, it is still 1:0, no goal so far.

Table 15: An illustrative example of the effectiveness of translation-based language model. The “Labels” column indicates whether the candidate responses are suitable to the query. The “Candidate Responses” and “Original Posts” columns list the candidate responses with their original posts, respectively. The “Before” and “After” columns show the ranking of responses in the top 30 candidates before and after using TransLM as one of the features, respectively.

Query	小舟划向夕阳（摄于西西里岛附近）。晚安！ A boat rowed off into the sunset (taken in the vicinity of Sicily). Good Night!			
Labels	**Candidate Responses**	**Original Posts**	**Before**	**After**
Suitable	真不错，夕阳西下，晚安！ Very beautiful sunset. Good night!	西西里岛的日落。大家晚安！ Sicily’s sunset. Good night!	1	2
Suitable	西西里岛的日落。大家晚安！ Sicily’s sunset. Good night!	西西里岛的日落。大家晚安！ Sicily’s sunset. Good night!	2	1
Suitable	日落夜不黑的欧洲夜晚 Sunset night, not black night in Europe	苏黎世日落。大家晚安！ Zurich sunset. Good night!	22	3
Suitable	种么美，想起《爱在黄昏日落时》的场景 What a beauty! I think of the “Love in the Sunset” scene	巴黎日落。大家晚安！ Paris sunset. Good night!	25	4

lexically dissimilar but semantically similar post-comment pairs, and rank them high. Table 15 gives some retrieved post-comment pairs for a given query. We can see that the model with TransLM as one of the features can rank the suitable responses higher than that without TransLM. This is due to the word translation probabilities: T(夕阳|日落)=0.018, T(夕阳|黄昏)=0.012, and T(西西里岛|欧洲)=0.001.

8.3. The Effectiveness of Deep Matching Model

Table 16 shows some retrieved responses for a given query. From the table, we can see that although the two suitable responses share almost no common words with the query, the model with DeepMatch as one of the features can match them well and rank them higher than that without the feature.

8.4. The Effectiveness of Topic-Word Model

Table 17 gives some retrieved responses for a given query. From the table, we can clearly see that the unsuitable responses, which do not share topic words with the query, are ranked lower when using TopicWord as one of the features; while the suitable response, which share topic words with the query, is ranked higher when using TopicWord as one of the features. More specifically, the word “菜鸟(rookie)” in the query is not a topic word, thus has a low term weight (i.e., the probability of being a topic word). Although the word “菜鸟(rookie)” in the unsuitable responses is

Table 16: An illustrative example of the effectiveness of deep matching model. The “Labels” column indicates whether the candidate responses are suitable to the query. The “Candidate Responses” column lists the candidate responses. The “Before” and “After” columns show the ranking of responses in the top 30 candidates before and after using DeepMatch as one of the features, respectively.

Query	洗车过程中就开始下雨，这是一种什么样的衰! It began to rain when I was washing the car. Such a bad luck!		
Labels	**Candidate Responses**	**Before**	**After**
Suitable	经常听朋友这么说。看来还是不洗的划算，哈哈! I often hear that from my friends. It is better not to wash it to save money, haha!	3	2
Suitable	同命人哇，你俩互相安慰安慰 You guys have the same situation, so at least you can comfort each other.	6	1

Table 17: An illustrative example of the effectiveness of topic-word model. The “Labels” column indicates whether the candidate responses are suitable to the query. The “Candidate Responses” column lists the candidate responses. The “Before” and “After” columns show the ranking of responses in the top 30 candidates before and after using TopicWord as one of the features, respectively.

Query	选用**代码控制工具** 的时候，还是用**SVN** ，不要用**GIT** ，虽然**GIT** 比**SVN** 强大很多。因为如果团队里有一个菜鸟，他各种各样关于**GIT** 的问题会烦死你。**SVN** 的优势就是极其简单，菜鸟也可以很快掌握 When choosing the **code control tool**, use **SVN** instead of **GIT**, although **GIT** is more powerful than **SVN**. The reason is that if a team has a rookie, he will bother you with all sorts of questions about **GIT**. The advantage of **SVN** is it is so simple that a rookie can grasp it quickly enough.		
Labels	**Candidate Responses**	**Before**	**After**
Unsuitable	我是**保密员** 、**菜鸟** 与**90后** **努力** 为了不成为老**菜鸟** I am a **confidential staff**, **rookie** and **the generation after 90s**, **making efforts** in order not to become an old **rookie**.	1	9
Unsuitable	我，还有很多用户。都是**菜鸟** 。 I and also a lot of users are **rookies**.	2	4
Suitable	一个**版本控制** 的工具 A **version control tool**	9	1

a topic word with a high term weight, the cosine similarities between the query and the two unsuitable responses are still not high after using the term weighting of topic words. Moreover, the suitable response is ranked higher mainly because it has common topic words “控制(control), 工具(tool)”, with the query, which are assigned higher weights.

8.5. Some Failed Issues

In this subsection, we show several issues which cannot be addressed well with our current model and we will leave them to future work.

8.5.1. Entity Association

Entity association is only partially addressed with features like query-response cosine similarity, with entity names treated as words, which is apparently not enough for preventing the following type of mistakes (see Table 18) when the candidate response and the query match well on other parts. Actually, for query 1, we only need to modify the word “李教授(Prof. Li)” in the unsuitable response 1 to the word “王教授(Prof. Wang)”, to get expected response 1, which is suitable to the query 1. For query 2, we only need to modify the word “画(drawing)” in the unsuitable response 2 to the word “瓷器(china)”, to get expected response 2, which is suitable to the query 2.

8.5.2. Logic Consistency

Our current model does not directly maintain the logic consistency between the candidate response and the query, since logic consistency requires a deeper analysis of the texts, and therefore hard to implement. Table 19 shows two examples which are semantically relevant, and correct with respect to speech act, but logically inappropriate.

Table 18: Two failed examples on entity association.

Query 1	**王教授**将会在下学期开一门自然语言处理的课程。 **Professor Wang** will give a course on natural language processing, starting next semester.
Unsuitable Response 1	羡慕啊。。我希望在未来的某段时间里我也可以参加**李教授**的课程。 Jealous.. I wish I can attend **Prof. Li**'s course too some time in the future.
Expected Response 1	羡慕啊。。我希望在未来的某段时间里我也可以参加**王教授**的课程。 Jealous.. I wish I can attend **Prof. Wang**'s course too some time in the future.
Query 2	台北故宫博物院精美**瓷器**展 The fine **china** from Exhibition at the National Palace Museum in Taipei
Unsuitable Response 2	这**画**看起来很不错。台北故宫博物院到处是国宝。 This **drawing** looks so nice. National Palace Museum in Taipei is full of national treasures.
Expected Response 2	这**瓷器**看起来很不错。台北故宫博物院到处是国宝。 This **china** looks so nice. National Palace Museum in Taipei is full of national treasures.

Table 19: Two failed examples on logic consistency.

Query 1	查了一下，王凤仪先生应该不是我的外祖父，虽然他们的事迹相似，并且同称「王大善人」。 I checked. Wang Fengyi is not my maternal grandfather, although they've done similar deeds and both are called "Wang the Well-doer".
Unsuitable Response 1	您外祖父是王凤仪先生啊 Wow, Wang Fengyi is your maternal grandfather.
Query 2	全球最薄最紧凑大屏旗舰智能手机Ascend P1s 首发是UMTS /HSPA + 版本，还会有中国移动TD 版本。 The debut of the world's thinnest, most compact, big-screen flagship smartphone Ascend P1s is UMTS / HSPA + version, there will be China Mobile TD version.
Unsuitable Response 2	有没有移动TD 版本的？ Will there be any version of China mobile TD version?

9. Conclusions

In this paper we have proposed a retrieval-based model for short text conversation (STC), to leverage massive data collected from social media. Our experiments show that the retrieval-based model performs reasonably well, when combined with a set of carefully designed matching features and a huge repository of conversation data.

This work opens to several interesting directions for future work with regard to STC. When performing retrieval-based STC, we need to consider matching between query and response in terms of semantic relevance. In addition, we may also need to consider matching between query and response in terms of speech act, sentiment, entity association, logic consistency and discourse structure. How to model these factors and how to enhance the accuracy based on the factors in STC are open and challenging issues.

References

Berger, A., Lafferty, J., 1999. Information retrieval as statistical translation. In: Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR '99. ACM, New York, NY, USA, pp. 222–229.

Chen, G., Tosch, E., Artstein, R., Leuski, A., Traum, D. R., 2011. Evaluating conversational characters created through question generation. In: FLAIRS Conference.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., Lin, C.-J., Jun. 2008. Liblinear: A library for large linear classification. Journal of Machine Learning Research 9, 1871–1874.

Herbrich, R., Graepel, T., Obermayer, K., 1999. Large margin rank boundaries for ordinal regression. Advances in Neural Information Processing Systems, 115–132.

Jafarpour, S., Burges, C. J., Ritter, A., 2010. Filter, rank, and transfer the knowledge: Learning to chat. Advances in Ranking, 10.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., Herbst, E., 2007. Moses: Open source toolkit for statistical machine translation. In: Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions. ACL '07. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 177–180.

Leuski, A., Patel, R., Traum, D., Kennedy, B., 2006. Building effective question answering characters. In: Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue. SigDIAL '06. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 18–27.

Leuski, A., Traum, D., 2011. Npceditor: Creating virtual human dialogue using information retrieval techniques. AI Magazine 32 (2), 42–56.

Li, H., 2011a. Learning to rank for information retrieval and natural language processing. Synthesis Lectures on Human Language Technologies 4 (1), 1–113.

Li, H., 2011b. A short introduction to learning to rank. IEICE Transactions on Information and Systems 94 (10), 1854–1862.

Li, H., Xu, J., 2014. Semantic matching in search. Foundations and Trends in Information Retrieval 7 (5), 343–469.

Litman, D., Singh, S., Kearns, M., Walker, M., 2000. Njfun: A reinforcement learning spoken dialogue system. In: Proceedings of the 2000 ANLP/NAACL Workshop on Conversational Systems - Volume 3. ANLP/NAACL-ConvSyst '00. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 17–20.

Liu, T.-Y., Mar. 2009. Learning to rank for information retrieval. Foundations and Trends in Information Retrieval 3 (3), 225–331.

Lu, Z., Li, H., 2013. A deep architecture for matching short texts. In: Advances in Neural Information Processing Systems. pp. 1367–1375.

Nouri, E., Artstein, R., Leuski, A., Traum, D., 2011. Augmenting conversational characters with generated question-answer pairs. In: Question Generation: Papers from the 2011 AAAI Fall Symposium.

Och, F. J., Ney, H., Mar. 2003. A systematic comparison of various statistical alignment models. Computational Linguistics 29 (1), 19–51.

Papineni, K., Roukos, S., Ward, T., Zhu, W.-J., 2002. Bleu: A method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. ACL '02. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 311–318.

Ritter, A., Cherry, C., Dolan, W. B., 2011. Data-driven response generation in social media. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. EMNLP '11. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 583–593.

Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M. M., Gatford, M., et al., 1995. Okapi at trec-3. NIST SPECIAL PUBLICATION SP, 109–109.

Salton, G., Wong, A., Yang, C. S., Nov. 1975. A vector space model for automatic indexing. Commun. ACM 18 (11), 613–620.

Schatzmann, J., Weilhammer, K., Stuttle, M., Young, S., 2006. A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. The Knowledge Engineering Review 21 (02), 97–126.

Voorhees, E. M., 2002. The philosophy of information retrieval evaluation. In: Evaluation of cross-language information retrieval systems. Springer, pp. 355–370.

Wang, H., Lu, Z., Li, H., Chen, E., October 2013. A dataset for research on short-text conversations. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Seattle, Washington, USA, pp. 935–945.

Weizenbaum, J., Jan. 1966. Eliza - a computer program for the study of natural language communication between man and machine. Communications of the ACM 9 (1), 36–45.

Williams, J. D., Young, S., 2007. Partially observable markov decision processes for spoken dialog systems. Computer Speech & Language 21 (2), 393–422.

Wu, W., Lu, Z., Li, H., Jan. 2013. Learning bilinear model for matching queries and documents. Journal of Machine Learning Research 14 (1), 2519–2548.

Xue, X., Jeon, J., Croft, W. B., 2008. Retrieval models for question and answer archives. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR '08. ACM, New York, NY, USA, pp. 475–482.

Zhai, C., Lafferty, J., 2001. A study of smoothing methods for language models applied to ad hoc information retrieval. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR '01. ACM, New York, NY, USA, pp. 334–342.

Zhang, H.-P., Yu, H.-K., Xiong, D.-Y., Liu, Q., 2003. Hhmm-based chinese lexical analyzer ictclas. In: Proceedings of the Second SIGHAN Workshop on Chinese Language Processing - Volume 17. SIGHAN '03. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 184–187.