Rich Short Text Conversation Using Semantic Key Controlled Sequence Generation

Kai Yu, Zijian Zhao, Xueyang Wu, Hongtao Lin and Xuan Liu

***Abstract*— With the recent advances of sequence-to-sequence framework, generation approaches for short text conversation (STC) become attractive. Traditional sequence-to-sequence approaches for short text conversation often suffer from poor diversity and general reply without substantiality. It is also hard to control the topic or semantics of the selected reply from multiple generated candidates. In this paper, a novel external memory driven sequence-to-sequence learning approach is proposed to address these problems. A tensor of external memory is constructed to represent interpretable topics or semantics. During generation, a controllable memory trigger is extracted given the input sequence, and a reply is then generated using the memory trigger as well as the sequence-to-sequence model. Experiments show that the proposed approach can generate much richer diversity than traditional sequence-to-sequence training with attention. Meanwhile, it achieves better quality score in human evaluation. It is also observed that by manually manipulating the memory trigger, it is possible to interpretably guide the topics or semantics of the reply.**

***Index Terms*— question and answer, chatbot, short text conversation, sequence to sequence learning**

I. Introduction

With the widespread usage of social media, such as Twitter and microblogs, in recent years, more and more open domain conversation data becomes feasible, which makes data-driven approach for conversation possible. *Short Text Conversation* (STC) [1] is a simplified conversation task: one round conversation formed by two short text sequences. It is widely used in conversation robot for chit-chat. The former one, usually given by human being, is referred to as a *post*, while the latter, given by computer, is referred to as a *comment*. The research on STC contributes to the development of open domain conversation system.

There are two major frameworks for short text conversation: retrieval based methods and generation based methods. Retrieval based methods [2] search the STC training corpus to find an existing comment which is most relevant to the post. Generation methods [3] [4] usually train a text generation model on the STC corpus and generate a comment using the model given a post. Compared to retrieval based methods, generation based methods can produce new comments that are not in the training set. This important feature makes generation methods very attractive.

The sequence-to-sequence model is one of the most prominent models for language generation [5] [6]. It consists of two parts, namely *encoder* and *decoder*. The encoder part encodes the variable length sequence into a fixed length vector. Then, the decoder part generates a variable length sequence from this vector word by word. Although this method successfully links variable length input and output into a single model, it suffers from vanishing gradient problem when the input is too long. In addition, a fixed length vector can not encode sufficient information when the input is long. Attention mechanisms [7] have been proposed to tackle this problem. When generating the next word, the decoder can access all hidden vectors of the encoder. Then, the decoder network decides which segment of the input is more relevant to the current situation by computing a soft alignment. The alignment is a byproduct of the sequence-to-sequence training.

Sequence-to-sequence training with attention mechanism has been successfully applied to machine translation [7], machine summarization [8] and even video caption generation [9]. However, direct use of sequence-to-sequence framework for short text conversation tends to generate short and dull responses [10] [11]. The response candidates in the beam for a given post are usually very similar and lack diversity. Meanwhile, it is not possible to control the semantics of the generated comments in traditional sequence-to-sequence framework. What the model learns is just the semantic mapping appeared in the parallel training data, and the model is not aware of any external knowledge. In this paper, a novel external semantic memory driven sequence-to-sequence framework is proposed to address the aforementioned problems. An external semantic memory, which contains rich response sentence embeddings associated with predefined semantic keys, is constructed before training. An external memory trigger vector is extracted from each post after the encoding process. The vector is then used, as an auxiliary feature, together with the post sentence embedding to be input to the decoder during training and generation. By enumerating different semantic keywords extracted from the post, it is possible to generate comments with rich diversity. Moreover, it is even possible to manually manipulated memory trigger process to introduce new semantics which does not exist in the post.

The rest of the paper is arranged as follows. Section II reviews sequence-to-sequence training approaches for STC. Section III proposes external memory driven sequence-to-sequence training, followed by the experiments in section IV. Then, the whole paper is concluded in section V.

II. Sequence-to-sequence Learning for STC

Sequence-to-sequence learning [5], also referred to as *encoder-decoder* framework [6], is firstly proposed in the

Kai Yu is the corresponding author. This work was supported by the National Key Research and Development Program of China under Grant No.2017YFB1002102 and the China NSFC projects (No. 61573241). Experiments have been carried out on the PI supercomputer at Shanghai Jiao Tong University.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2018.2819941, IEEE/ACM Transactions on Audio, Speech, and Language Processing

field of machine translation. The method employs a *recurrent neural network* (RNN), usually *long short term memory* [12] network (LSTM) or *gated recurrent unit* [6] (GRU), to map the input sequence to a vector of a fixed dimensionality, and then another RNN to decode the target sequence from the vector. GRU is used in this paper as it achieves comparable performance against LSTM with less computation [13]. A GRU is defined as:

$$\begin{aligned} \boldsymbol{r}_t &= \sigma(\boldsymbol{W}_r \boldsymbol{x}_t + \boldsymbol{U}_r \boldsymbol{h}_{t-1}) && (1)\\ \boldsymbol{z}_t &= \sigma(\boldsymbol{W}_z \boldsymbol{x}_t + \boldsymbol{U}_z \boldsymbol{h}_{t-1}) && (2)\\ \tilde{\boldsymbol{h}}_t &= \tanh\left(\boldsymbol{W} \boldsymbol{x}_t + \boldsymbol{U}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1})\right) && (3)\\ \boldsymbol{h}_t &= \boldsymbol{z}_t \odot \boldsymbol{h}_{t-1} + (1 - \boldsymbol{z}_t) \odot \tilde{\boldsymbol{h}}_t && (4) \end{aligned}$$

where $\boldsymbol{x}_t$ is the input word embedding vector at time t, $\boldsymbol{h}_t$ is the output hidden state vector, $\boldsymbol{r}_t$ is the output of the reset gates, $\boldsymbol{z}_t$ is the output vector of the update gate, $\boldsymbol{W}$ and $\boldsymbol{U}$ are weight matrices and $\odot$ denotes element-wise product. From the above formula, GRU can take in a variable-length sequence of vectors $\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T\}$ and convert them to a fixed-length vector $\boldsymbol{h}_T$. This recurrent calculation characteristics makes it suitable for sequence modelling.

In sequence-to-sequence training, the encoder part is usually a standard GRU to map a sequence into a fixed-size sentence-level context vector. In contrast, the decoder takes the context vector as an additional input and generates the output sequence word by word. The formula of the decoder GRU are similar to equations (1) to (3) except for including the context vector as additional input. For example, given $\boldsymbol{h}_T^{(\mathsf{e})}$ as the encoder output vector, equation (1) of the decoder can be rewritten as :

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r \boldsymbol{x}_t + \boldsymbol{U}_r \boldsymbol{h}_{t-1} + \boldsymbol{C}_r \boldsymbol{c}_t) \quad (5)$$

where $\boldsymbol{C}$ is weight matrix of context vector and $\boldsymbol{c}_t = \boldsymbol{h}_T^{(\mathsf{e})}$ is the sentence-level context vector.

A widely-used extension of the encoder-decoder framework is to employ the *attention* mechanism [7], which replaces the constant context vector $\boldsymbol{h}_T^{(\mathsf{e})}$ with a time-dependent context vector. The time-dependent context vector $\boldsymbol{c}_t$ is constructed using weighted sum of all hidden state vectors of the encoder. The weights can be seen as a distribution to describe the relevance between all encoder hidden state vectors and the current hidden state vector of the decoder. There are many ways to compute the context vector and a popular implementation [14] is used in this paper:

$$e_{ti} = \boldsymbol{h}_t^{(\mathsf{d})\top} \boldsymbol{W}_c \boldsymbol{h}_i^{(\mathsf{e})} \qquad \alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{T} \exp(e_{tk})} \quad (6)$$

$$\boldsymbol{c}_t = \sum_{i=1}^{T} \alpha_{ti} \boldsymbol{h}_i^{(\mathsf{e})} \quad (7)$$

where $\boldsymbol{h}_t^{(\mathsf{d})}$ and $\boldsymbol{h}_t^{(\mathsf{e})}$ are hidden state vectors at time t of the decoder and the encoder respectively, T is the total length of the encoder input sequence and $\boldsymbol{c}_t$ is final context vector at time t. Sequence-to-sequence learning with attention mechanism has already been successfully applied to many tasks, such as machine translation [7], question answering [15] and so on. It has also been applied to short text conversation (STC) [3]. An example is depicted in Fig. 1.

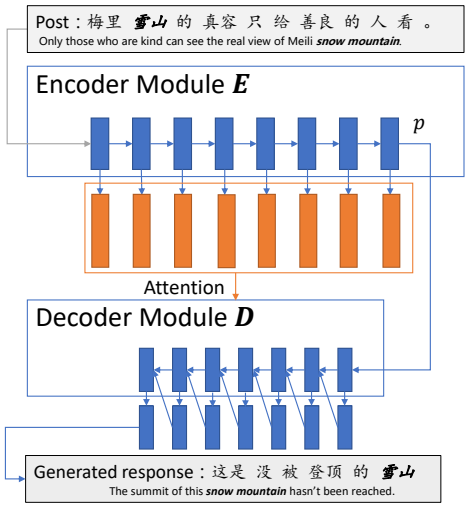

Fig. 1: Sequence-to-sequence Learning with Attention for Generation-based STC

However, when used for STC, the above encoder-decoder framework tends to generate short and dull responses. This phenomenon is due to the nature of conversation. Different from machine translation, where source sentence and target sentence express the same meaning in two different languages, or machine summarization, where the summary text is basically a fragment of the source document, the response for a post in STC would introduce additional information that may not appear in the post and require external knowledge to understand. What's more, the additional information can be arbitrary, as long as it is relevant to the post. Thus, there might exist multiple suitable responses for a single post. In contrast, target text almost only varies at the surface level, i.e. lexicon or grammar, for machine translation or machine summarization.

Due to the semantic distinction, the comment distribution of short text conversation is very complex and highly variable. It is hard for a sequence-to-sequence model to accurately capture the semantic mapping from post to comment, as external knowledge may be required for this kind of mapping. Moreover, the cross entropy training criterion, which optimizes the KL divergence between the true data distribution and the model distribution, tends to fit all modes of the data distribution. This tendency makes the mode of the model distribution appear in a less-likely area when the model capacity is lower than the complexity of the data distribution. Meanwhile, compared to a language model trained on comments, the perplexity reduction from a sequence-to-sequence model is smaller. This phenomenon means that a post may not provide enough information to generate the corresponding comment. Therefore, general responses without substantiality may get high probability and be frequently generated.

[16] proposes to modify the basic encoder-decoder framework to generate high-quality and informative conversation responses. A self-attention mechanism is added to the decoder

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2018.2819941, IEEE/ACM Transactions on Audio, Speech, and Language Processing

to maintain coherence in longer responses, and a stochastic beam-search algorithm with segment-by-segment reranking is introduced to inject diversity earlier in the generation process. Although this type of strategy is effective in generating longer and informative responses, it can not ease the problem we describe above totally. A general approach to solve the dull response problem is to provide additional information to the decoder. [17] utilizes a topic model to extract relevant topic words of the post. These extracted topic words are meaningful and considered to be very likely to appear in the responses. The decoder would focus on the additional topic words and is encouraged to generate the additional topic words. In addition to word-level supplementary information, [18] and [19] exploit sentence-level supplementary information and combine the generation model with a retrieval-based model. [18] retrieves the most similar post in the corpus and takes the corresponding response as another input to the decoder. [19] preserves relevant external facts about the entity in the conversation history in a memory network. The decoder then pays attention on the memory and automatically select the useful information for generation.

Although these methods enhance the model and produce better results due to the additional information, these methods are not controllable. What's more, when beam search is used during decoding, the diversity of candidates in a beam are known to be small. Consequently, the generated responses candidates of the above methods are usually similar and differences mainly lie at surface level. The inherent reason for lacking diversity is that these methods ignore the fact that there exist multiple suitable responses for the same post. When we chat with others, we understand what the other person says first, then decide what to reply, and express it in nature language. The second step involves external knowledge, such as common sense, background knowledge and personal preference. This is the most important factor for diversity of the responses. An ideal conversation system should imitate the controlled generation process.

There have been a few relevant works addressing this issue. The method proposed in [19] retrieves some "facts" given the post and then takes them supplementary materials for generation. The idea of introducing external data related to the post is very similar between [19] and our work. However, our work focus on external semantic memory construction to better use external data while [19] concentrates more on exploitation of different variants of the multi-task system. For the controllability of conversation, [20] and [21] study controllable sequence-to-sequence frameworks for conversation and achieve interesting results. However, they can only control emotion or tense rather than semantics which is essential in the STC task.

In order to imitate the process of conversation, we should find the semantics of the response before generating the final responses. However, there is no well designed semantic representation for open domain conversation. [22] [23] [24] adopt a two step generation framework and regard keyword(s) in response as an alternative semantic representation for response. [22] first generates a keyword according to point mutual information, then uses the backward-forward method to make the keyword appear in the response. Similar to [22], [23] propose an implicit content-introducing method which utilize the auxiliary cue word information implicitly through a hierarchical gated fusion unit. The cue words are incorporated into the generation process but they do not necessarily appear in the response. In contrast, [24] first generates a sequence of keywords, which is the noun sequence of the response in training, then generates the final response according to the post and the noun sequence. In these models, the semantic representation are the exact keyword(s). This representation is very limited. What we actually care about are not the exact keywords, but the reply contents which are linked to these keywords. Also, these methods can not fully solve the diversity problem since multiple keyword sequences from beam search are still similar.

III. External Memory Guided Sequence-to-Sequence Learning

In this work, we combine the advantages of [19] and [22] and propose a new sequence-to-sequence learning approach for STC. A tensor, in the form of a list of matrices, is constructed to represent the semantics of the comment sentences, referred to as *external semantic memory*. Each matrix represents all possible comment sentences corresponding to a specific *semantic key*. Each row vector of the matrix forms a sentence embedding basis and all row vectors span the whole comment semantic space of the specific semantic key. During generation, a semantic key is extracted from the input sequence and used to construct a comment sentence embedding from the external memory. The final comment is then generated using the embedding from external memory as well as the post sequence embedding with a sequence-to-sequence model. By manipulating the semantic keys, it is possible to interpretably guide the topics or the semantics of the comment.

A. General Framework

This *External Semantic Memory Guided Sequence-to-Sequence Learning* framework consists of three components: an encoder $\boldsymbol{E}$, a decoder $\boldsymbol{D}$ and an external memory $\boldsymbol{M}$. Fig. 2 depicts the general framework and the information flow.

An *external memory* is introduced into the standard encoder-decoder framework, which is assumed to be constructed using large amount of data (possibly unsupervised) outside the training dataset. Hence, $\boldsymbol{M}$ can be regarded as an explicit storage of external knowledge, which enables the encoder-decoder model to use knowledge outside the post or even training data. The usage of the external memory is similar to how we chat with each other. We usually focus on some specific semantic keys (e.g. keyword or topic) of a sentence and then think about the related semantics given our goal or background knowledge (memory), and finally compose a response sentence based on the newly composed output semantics. We simulate this process using the external semantic memory module. First, the external semantic memory is constructed and indexed using semantic keys. When generating a response, we first get the output semantic key and address the memory blocks to obtain the embedding for the output. Finally, the decoder model

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2018.2819941, IEEE/ACM Transactions on Audio, Speech, and Language Processing

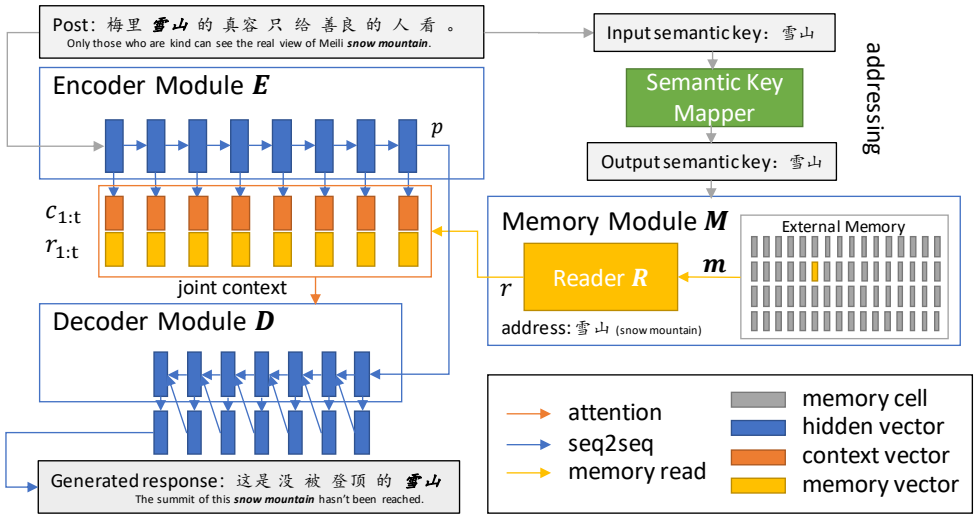

Fig. 2: General Framework of Controllable Short-Text-Conversation Generation with External Memory

generates the comment based on both the input post embedding and the extracted output sentence embedding from the external memory. The whole framework of encoder-decoder with external memory is formulated as below.

Let $\boldsymbol{x}_{1:T} = \{x_1, x_2, ..., x_T\}$ represent the embeddings of words in post, where T represents the length of post. The encoder module $\boldsymbol{E}$, receives word embeddings produces a dense representation $\boldsymbol{p}$ of the input post sentence and a set of context vectors $\boldsymbol{c}_{1:T} = \{c_1, c_2, ..., c_T\}$ (c for short), in equation (8). In this paper, we simply use a uni-directional GRU as the encoder.

$$\boldsymbol{p}, \boldsymbol{c}_{1:T} = \boldsymbol{E}(\boldsymbol{x}_{1:T}) \tag{8}$$

We then extract the semantic keys from the input sentence. A typical semantic key is just a word or a phrase, although a distributed representation of the topic can also be used (which will be discussed later). In this paper, only nouns, adjectives, and idioms are chosen as the semantic keys. The LTP tool[1] [25] for POS tagging is used in this step. Usually, multiple semantic keys can be extracted from the post, referred to as the *input semantic key*. We may choose anyone of them to guide further comment generation process. By manipulating the input semantic key, such as iteratively or randomly choose one input semantic key, we may generate diverse comments with different semantic preferences. Once an input semantic key $\boldsymbol{k}^{(\mathrm{i})}$ is found, it is fed into a semantic key mapper $\boldsymbol{S}$ to be converted to the output semantic key $\boldsymbol{k}^{(\mathrm{o})}$:

$$\boldsymbol{k}^{(\mathrm{o})} = \boldsymbol{S}(\boldsymbol{k}^{(\mathrm{i})}) \tag{9}$$

The semantic key mapper $\boldsymbol{S}$ can be trivial, e.g. directly using the input-side semantic key as the reply-side semantic key, i.e. $\boldsymbol{k}^{(\mathrm{o})} = \boldsymbol{k}^{(\mathrm{i})}$. Alternatively, it can be complex, e.g. using a synonym, or a word with nearest embedding, or a word inferred from knowledge graph etc. In case of using a topic vector as the semantic key, the external memory topic vector closest to the post topic vector can be chosen. More aggressively, it is even possible to manually make up an output semantic key regardless of the input and force the latter generation process to use it. This will be discussed in the experiment section. In summary, the semantic key mapper enables controllability of the proposed model.

After determining the output semantic key, we need to use it to index the external semantic memory $\boldsymbol{M}$. $\boldsymbol{M}$ consists of K memory blocks, where K is the number of all possible output semantic keys. Each output semantic key corresponds to an address, or index, of a memory block. Each memory block is a $L \times D$ matrix $\boldsymbol{m}$ associated with a particular semantic key, where L is the number of rows representing the number of the output sentence embedding bases, and D denotes the dimensionality of the sentence embedding. According to memory construction process (described below), D is the same as the dimensionality of the input post embedding. A memory reader $\boldsymbol{R}$ conducts reading process on the selected memory block $\boldsymbol{m}$ to produce an external memory context vector $\boldsymbol{r}$

$$\boldsymbol{r} = \boldsymbol{R}(\boldsymbol{m}, \boldsymbol{p}) \tag{10}$$

Inspired by Neural Turing Machine [26], we apply a content-based addressing to the matrix of the chosen memory cell. Given a post p, the reader R will return a weighted-summed vector on the matrix $\boldsymbol{m}$ following equation (11), where w_i is given by equation (12) and $\boldsymbol{m}(i)$ denotes the i^{th} row in matrix $\boldsymbol{m}$. β in equation (12) is a coefficient to control the sharpness of the weight vector [27]. To ensure the sharpness, the value of β is set to 100 in this paper.

$$\boldsymbol{r} = \boldsymbol{R}(\boldsymbol{m}, \boldsymbol{p}) = \sum_{l=1}^{L} w_l \boldsymbol{m}(l) \tag{11}$$

$$w_l = \frac{\exp(\beta\, \boldsymbol{p}^\top \boldsymbol{m}(l))}{\sum_j \exp(\beta\, \boldsymbol{p}^\top \boldsymbol{m}(j))} \tag{12}$$

[1] https://github.com/HIT-SCIR/ltp

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2018.2819941, IEEE/ACM Transactions on Audio, Speech, and Language Processing

After obtaining the memory context vector $\boldsymbol{r}$, we duplicate it for each input word instance and append it to the original context $\boldsymbol{c}_{1:T}$ to form a new joint context vector set $\widetilde{\boldsymbol{C}}$.

$$\widetilde{\boldsymbol{C}} = \{[\boldsymbol{c}_1; \boldsymbol{r}], [\boldsymbol{c}_2; \boldsymbol{r}], ..., [\boldsymbol{c}_t; \boldsymbol{r}]\} \tag{13}$$

During decoding, the decoder $\boldsymbol{D}$ generates the reply comment (denoted as $\boldsymbol{y}$) using context vector from both the post sentence and the external semantic memory. In this paper, the decoder is a uni-directional GRU with attention mechanism on the joint context.

$$\boldsymbol{y} = \boldsymbol{D}(\boldsymbol{p}, \widetilde{\boldsymbol{C}}). \tag{14}$$

The whole process of the external semantic memory guided sequence-to-sequence learning is summarized in algorithm 1. It is worth noting that external memory construction is independent of the encoder-decoder training. Given an external memory, standard gradient descend algorithm can be used to train the encoder-decoder model (with the memory matrix $\boldsymbol{m}$ fixed) to maximize the sequence generation likelihood $P(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{r})$.

Algorithm 1 External Semantic Memory Guided Sequence-to-sequence Generation

Receive an input post
Convert the post to a word embedding sequence $\boldsymbol{x}_{1:T}$.
Extract the input semantic key $\boldsymbol{k}^{(\mathtt{i})}$
Map it to the output semantic key $\boldsymbol{k}^{(\mathtt{o})} = \boldsymbol{S}(\boldsymbol{k}^{(\mathtt{i})})$
Find the associated memory block matrix $\boldsymbol{m}$
Encode the post using the encoder
$\boldsymbol{p},\ \boldsymbol{c}_{1:T} = \boldsymbol{E}(\boldsymbol{x}_{1:T})$
Read the memory context vector from the external semantic memory
$\boldsymbol{r} = \boldsymbol{R}(\boldsymbol{m}, \boldsymbol{p})$
Append $\boldsymbol{r}$ to the original encoder context vectors
$\widetilde{\boldsymbol{C}} = \{[\boldsymbol{c}_1; \boldsymbol{r}], [\boldsymbol{c}_2; \boldsymbol{r}], ..., [\boldsymbol{c}_t; \boldsymbol{r}]\}$
Decode to generate the comment
$\boldsymbol{y} = \boldsymbol{D}(\boldsymbol{p}, \widetilde{\boldsymbol{C}})$

B. External Semantic Memory Construction

The previous section describes the general framework of the proposed approach. In this section, external semantic memory construction is introduced in detail.

The content of each memory block is a matrix including K representative comment sentence embeddings corresponding to a specific output semantic key. The "semantic key" can be represented by a keyword (one-hot vector) or a topic embedding vector. The address of each memory block is the index of the associated output semantic key. In case of keyword based representation, it is the keyword index of the one-hot vector. In case of topic embedding vector, vector quantization is applied first to discretize the topic embedding space. The semantic key index is then the index of the resultant code book. As shown in equation (9), output semantic keys can be obtained by mapping the input semantic keys using the semantic key mapper $\boldsymbol{S}$.

The external semantic memory is constructed independent of the training process of the encoder-decoder described in section III-A. This separation allows us to incorporate information outside the encoder-decoder training data. Both the employed algorithm as well as the used data can be different from the main encoder-decoder training. Since each memory block is effectively a collection of sentence embedding vectors, an encoder model is required. In this paper, depending on the data for external memory construction, two different approaches are used to find the encoder:

- **Encoder-decoder**. Given parallel STC data where both posts and comments are available, standard sequence-to-sequence learning (with or without attention) can be pre-trained. Here, the data can be different from the data for the external semantic memory guided training to incorporate external knowledge, but the data from similar corpus are preferred. Once the training is done, the decoder is discarded and only the encoder is used to convert all comments in the training data into sentence embeddings. Ideally, we want to encode the knowledge of the post-comment pairs and the relationship between posts and comments. We believe that using the encoder trained from post-comment pairs to encode comments can somehow capture the knowledge and relationship. Additionally, it is also compatible with the generation process since the external semantic memory is used in encoder side.
- **Auto-encoder**. When post-comment pairs are not available, it is also possible to use large amount of non-parallel sentences such as news, novel, or other text materials to incorporate richer external knowledge. Here, auto-encoder is trained on the non-parallel data set and convert the data into sentence embeddings.

The sentence embeddings obtained using the above encoders are regarded as the output comment embeddings. To build the external semantic memory, we need to associate the comment embeddings to output semantic keys. With the same semantic key extraction approach as previously described, we can obtain semantic keys for each comment embedding. Then, these sentence embeddings can be grouped together according to their semantic keys. Note that each sentence may have multiple semantic keys. Hence, it is possible that one sentence embedding may be put into multiple semantic key groups. Once the grouping is done, we need to construct a fixed-size memory block for each semantic key, i.e. select the K most representative sentences from all sentences inside each group. To achieve this, we first use truncated SVD [28] to perform latent semantic analysis (LSA) [29] on all sentence embeddings in the same memory block and obtain the projection vectors in the LSA space. K-means algorithm [30] is then used on the projected vectors to form K clusters. Finally, we select one sentence embedding (original embedding space) for each cluster center whose corresponding latent projection vector is closest to the center. In the case of the memory block that has less than K sentences, we pad it with zero vectors.

C. Encoder-decoder Training with External Semantic Memory

With the constructed external memory, the training procedure can be divided into two parts: data pre-processing and end-to-end encoder-decoder training.

1) Data Pre-processing: As mentioned above, one sentence may be put into multiple semantic key groups in the memory construction. The training data is also processed under the same rule. As the training corpus is post-comment pairs (i.e. input and output sentences), we filter the training pairs according to their semantic keys. This filtering has two advantages: 1) more convenient for mini-batch training; 2) likely to filter out some generic responses.

First we extract semantic keys of the input sentence, then the semantic key mapper converts these keys into related output semantic keys. The training pair will then be put into different key groups based on the output semantic keys. It is worth noting that one input post sentence may correspond to multiple output comment sentences. By applying the grouping, we semantically classify the post-comment pair according to their output sentences, and remove many generic dull reply comments.

2) End-to-End Training and Generation: The training process follows the description in section III. During training, the external semantic memory is fixed, hence it is essentially an extra input to the decoder. The rest components of the model is a standard encoder-decoder structure with attention as shown in Fig. 2. We randomly select one reply-side semantic key group from the training data, and then construct the mini-batch. Note that, for each sample pair, we have already known its external memory address, i.e. relevant comment-side semantic key. The objective function for the encoder-decoder is defined as

$$\mathcal{L} = \sum_{t=1}^{T} \log p(\boldsymbol{y}_t | \boldsymbol{y}_{1:t-1}, \boldsymbol{x}, \boldsymbol{k}^{(\mathtt{o})}) \tag{15}$$

The objective function is fully differentiable w.r.t. the neural network parameters. During the generation process, semantic keys are first extracted from the post. If there are multiple input semantic keys, they are used in turn to generate multiple comment candidates. These candidates are then sorted according to their likelihood to yield the final top-1 or top-N output.

IV. Experiments

We use roughly 4.2 million pairs of STC data from Weibo (Chinese version of Twitter) for encoder-decoder training. The data set comes from the first and second STC challenges and detailed descriptions can be found in [31] and [32]. The LTP tool is used for word segmentation and POS tagging. The same vocabulary of size 48773 is used for both posts and comments covering more than 98% words. All the remaining words are mapped to a special token "UNK". Then the data set is filtered as stated in section III-C.1 which reduces its size to 1.5 million. All the data are used for external memory construction while only 1.2 million pairs out of the data set is used for training to show the potential of utilizing external corpora. 1000 pairs are held out from the data set as the test set for objective evaluation.

We built three types of sequence generation systems: the *neural responding machine* (NRM) [3] (i.e. the standard sequence-to-sequence training with attention), the *multi-resolution recurrent neural network* (MrRNN) [24] and the proposed external semantic memory guided encoder-decoder model (ESED). For our model, we used three ways to construct the external memory: the basic sequence to sequence model (S2S) [5], the sequence-to-sequence with attention (Atten) [7] and an auto-encoder (AutoED) trained only on the comments (this can be seen as utilizing the non-parallel corpus).

The encoder and decoder structures of ESED are the same. It is a 1-layer GRU with 400-dimensional word embedding and 800 dimensional hidden state vector. Nouns, adjectives and idioms are used as the semantic keys. All parameters are initialized with a uniform distribution between -0.05 and 0.05. Adam optimizer [33] is used with an initial learning rate of 0.0004. The mini-batch size is set to be 64. For NRM, we use the same dimensions of word embedding and hidden states as our model. For MrRNN, we replace the HRED (hierarchical recurrent encoder-decoder) model with LSTM and utilize both nouns and adjectives as key words. All of these models are trained multiple epochs to achieve the best valid perplexity. Beam search is used during generation. It is worth noting that for ESED, multiple input semantic keys are used in turn to generate multiple comment candidates. We rank the candidates in descending order of the length of semantic key because we think the longer word contains much more information.

A. Diversity and Substantiality Analysis

In this paper, we propose two objective analysis measures for STC: diversity and substantiality. Both of them can be easily and directly computed from the generated comments without knowing the reference comments. This avoids the problem that the coverage of references may be very low in the comment text space.

- **Diversity** reflects the richness of the words in the generated comments and defined as

$$\mathtt{Div} = \frac{\#(unique\ words)}{\#(all\ words)} \tag{16}$$

  The calculation is done for all test sentences by counting the total number of words and the number of uniquely appeared words. The basic assumption here is that the richer the generated vocabulary is, the more diverse the generated comments are. From the metric, it can be easily seen, general replies without substantiality will reduce the diversity metric.
- **Substantiality** reflects the substantial information contained in the generated responses, which is defined as the number of meaningful entities. We do not use name entity recognition (NER) tools to extract entities because these tools usually only identify people names, place names and organization names, which is relatively limited. Instead, we use maximum string matching algorithm based on

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2018.2819941, IEEE/ACM Transactions on Audio, Speech, and Language Processing

the Wikipedia entity table[2] to identify meaningful entities. We define substantiality as the average entity word number per sentence on the test set:

$$\texttt{Sub} = \frac{\#(entity\ words)}{\#(sentences)} \tag{17}$$

TABLE I: Diversity and Substantiality Analysis

Model	Ext. Mem.	Div		Sub	
		Top-1	Top-5	Top-1	Top-5
NRM	—	0.157	0.042	0.478	0.475
MrRNN		0.184	0.076	0.360	0.398
ESED	S2S	0.228	0.086	0.553	0.468
	Atten	0.232	0.088	0.520	0.460
	AutoED	0.227	0.080	0.540	0.478

For each post in the test set, we select top-1 and top-5 comments to calculate these two metrics. The results are shown in Table I. ESED consistently outperforms the two baseline models both in diversity and substantiality, and the difference is statistically significant for top-1 results. For top-5 results, the diversity of ESED decreases less than the NRM model which tends to generate similar and dull responses. This shows the ability of ESED to generate more diverse and rich comments. The reduction of substantiality from top-1 result to top-5 result may be due to the reduction of information contained in the semantic keys (longer semantic key is ranked higher).

TABLE II: Controllable Short Text Conversation Examples

Post	李娜太牛了!中国第一个世界网球大满贯冠军! Li Na is great! China's first World Tennis Grand Slam champion!	
NRM	李娜是世界冠军! Li Na is the world champion!	
MrRNN	李娜加油,加油! Come on, Li Na. Come on!	
ESED	Key-Word	Response
	世界 world	李娜是世界上最棒的! Li Na is the best in the world!
	中国 China	李娜是中国的骄傲! Li Na is the pride of China!
	网球 tennis	中国网球公开赛,加油! China Tennis Open, come on!
Post	感人器官捐献广告,请感谢那些给你第二次生命的人。 A touching organ donation advertisement, please thank those who gave you a second life.	
NRM	给我第二次生命的人。 A man who gives me a second life.	
MrRNN	感谢每一个人! Thanks for everyone!	
ESED	Key-Word	Response
	生命 life	感谢生命给予的一切! Thanks for everything given by life!
	器官 organ	感谢那些器官捐赠者。 Thanks for those organ donors.
	广告 advertisement	很感人的广告! A very touching advertisement!

We believe the above performance gains of ESED is mainly contributed by the use of semantic keys. Especially when the training data for ESED is partitioned into post-comment data pairs sharing the same semantic key, we are effectively removing generic responses. To get a concrete idea of this, some examples are given in Table II. From these examples, it can be observed that both NRM and MrRNN tend to sensible, safe but insubstantial answer, whereas ESED can use different semantic key to generate more concrete and diverse comments although some semantic keys can lead to unrelated comments.

The ESED model in Table I uses words (noun, adjective and idiom) as the representation of semantic keys. It is also of interest to compare this word-level semantic key to topic-level semantic key.

Regarding the topic model used in our experiments, we seek to train such a model that fit into the short text scenario. Comparing to LDA [34], Biterm Topic Model [35] (BTM)[3] is more effective at modeling short texts. Also, the model should cluster the texts in a proper granularity: neither too general (uninformative) or too specific (overfitting). Thus we choose the different topic sizes (200 and 500) in experiments. The detailed training procedure and settings are the same as the original paper referred as above. We train the topic models as described above first and assign each sentence a unique topic by choosing the topic with maximum probability. We assume that a trivial identity semantic mapper is used for topic memory. Hence, we only need to estimate the topic for a post and the semantic key index is just the index for the topic. To investigate the effect of topic numbers, we set the number of topics to 200 and 500 respectively. The diversity and the substantiality performance of topic-level memory is shown in Table III. For comparison, the comparable word-level memory result is also listed.

TABLE III: ESED with Topic-level External Memory

Model	Ext. Mem.	Div		Sub	
		Top-1	Top-5	Top-1	Top-5
ESED	Word	0.232	0.088	0.520	0.460
	T200	0.217	0.063	0.397	0.406
	T500	0.211	0.061	0.389	0.386

It can be observed that topic-level external memory performs a lot worse than word-level memory. We believe this is largely because the semantics of a topic is so vague that we can not control the semantics of the generated comment well. Also, there are no explicit relations between two topics, hence it is also hard to define a meaningful semantic key mapper. What's more, since each sentence can only have one topic, it is more difficult to post-comment pairs sharing the same topic for ESED training. In fact, we observed that the number of data pairs for training ESED with topic-level memory is significantly smaller. All these contribute the degradation in Table III. In the rest of the paper, we only focus on word-level external memory.

B. Objective Evaluation of Reply Quality

In this section, we use BLEU score w.r.t. the reference comment to calculate an objective measure to evaluate the reply comment quality. The BLEU score [36] was first proposed to evaluate the quality of machine translation. Recently, it has also been used as objective indicator of reply quality in dialogue system. The results are shown in Table IV.

It can be observed that for both top-1 and top-5 generated comments, ESED achieves better BLEU scores than the baseline models. Although it has been argued that the consistency of BLEU score with human evaluation is poor when it is used for dialogue system evaluation [37], consistent performance gains still demonstrate the effectiveness of the proposed ESED approach.

[2]https://www.wikidata.org/wiki/Wikidata:Main_Page

[3]https://github.com/xiaohuiyan/BTM

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2018.2819941, IEEE/ACM Transactions on Audio, Speech, and Language Processing

TABLE IV: BLEU Score Comparison

Model	Ext. Mem.	BLEU	
		Top-1	Top-5
NRM	—	10.8	10.9
MrRNN		9.0	7.4
ESED	S2S	11.7	11.7
	Atten	11.2	11.9
	AutoED	13.2	13.2

C. Human Evaluation of Reply Quality

We also conduct human evaluation to compare the reply comment quality of different models on the top-1 setup. We follow the evaluation criterion of STC-2 challenge [32]. The appropriateness of replies is judged from the following four criteria:

(1) **Fluent**: the comment is acceptable as a natural language text;
(2) **Coherent**: the comment should be logically connected and topically relevant to the original post;
(3) **Self-sufficient**: the assessor can judge that the comment is appropriate by reading nothing other than the post-comment pair;
(4) **Substantial**: the comment provides new information in the eye of the originator of the post.

If either (1) or (2) is untrue, the comment should be labeled "L0"; if either (3) or (4) is untrue, the label should be "L1"; otherwise, the label is "L2". And in order to emphasize the problem of dull responses (i.e. generic comment), we add a special symbol "LD" to label reasonable but dull responses. And when we compute the mean score, "L2" is 2 points, "L1" is 1 point, "L0" is 0 point. Specially, we count "LD" as 0 points to encourage the system to generate diverse and informative responses instead of universe responses. The final average score is just the mean value of the scores from all speakers on all test sentences.

TABLE V: Human Evaluation Results

Model	Ext. Mem.	LD	L0	L1	L2	Ave.
NRM	—	10.7%	60.5%	16.3%	12.5%	0.41
MrRNN		4.8%	63.3%	19.7%	12.2%	0.44
ESED	S2S	5.0%	58.5%	17.2%	19.3%	0.56
	Atten	6.3%	57.8%	16.5%	19.3%	0.55
	AutoED	7.8%	55.8%	19.2%	17.2%	0.54

Our test set consists of 100 posts held out from the training set. Six annotators score the generated responses according to the above criteria. The results are shown in Table V. For LD/L0/L1/L2, the number is the percentage of generated sentences, Ave is the mean score. It is clear that ESED models generate much less generic dull responses than the baseline NRM models. Although ESED have larger LD percentage than MrRNN, this might be because MrRNN tends to generate many insensible answers. In general, ESED models can generate more coherent and informative comments which are appreciated by human. This is also consistent with the conclusion in Table I.

D. Analysis of Semantic Key Guidance

The previous sections have shown the effectiveness of the ESED approach for STC. These experiments all employ a trivial identity semantic key mapper function. The rich diversity of the generated comments mainly come from multiple semantic keys extracted from the same post. However, as stated before, one major advantage of ESED is that it is possible to control the generated semantics by manipulating the semantic key mapper. In this section, we will discuss this aspect.

1) Controllable Semantic Key Mapping Methods: In addition to using identity function as the mapper, there are many other *automatic* ways to map the input semantic keys. The idea is to find words or phrases semantically related to the input semantic keys. Either *linguistic mapping*, such as synonym or antonym, or *data driven mapping*, such as word embedding [38] neighborhood, can be used. Besides the automatic ways, it is also possible to manually control the mapping, i.e. set the output semantic key. Table VI shows the generated comments of the same post using different semantic mapping functions.

TABLE VI: Comments Generated Using Different Semantic-key Mapping Methods

Post	美好的社会应该由善良的人组成。 A good society should be composed of people of kindness.	
Mapping-Method	Key-Word	Response
Identity Mapper	善良 kindness	善良是一种信仰。 Kindness is a belief.
Synonym	友善 friendliness	善良是一种美德。 Kindness is a virtue.
Antonym	丑恶 evil	没有丑恶的社会,更可怕。 A society without evils is more terrifying.
Embedding	乐观 optimism	乐观是一种生活态度。 Optimism is an attitude of life.
Common-Reply	真 really	真的吗? Really?
	对 right	对的,对的。 Right, right.
Random	人 person	每个人都有善良的一面。 Every person has a side of kindness.
	中国 China	中国的贫富差距太大了。 The gap between the wealthy and the poor in China is too big.

It can be observed that with a synonym or antonym mapping function, the generated comments have close semantic link to the post but different content. This semantic richness demonstrates the advantage of using external memory. The "common-reply" semantic keys are the key words correspond to over ten thousands post-comment pairs. We believe these keys are signs of common replies which should lead to dull response as shown in Table VI. We also use a randomly selected semantic key to show the possibility of guiding the generated comments. As the table shows, most of these comments are capable of keeping semantic relevance and topic coherency. However, it is worth noting that not all semantic mapping function will lead to sensible results. We have also observed that when using data driven mapping function like embedding or random mapping, the systems may generate incomprehensible comments. Using some common keys can also lead to dull comments. Usually, identity or linguistic mapping are more reliable.

2) Knowledge Introduction by External Memory: Besides of the diverse and informative responses, our model has the potential to introduce new knowledge into the generated comments due to the use of external memory. Considering the external semantic memory construction procedure where

the content of a memory block is a matrix including the top-K representative comment sentence embeddings, it is likely that knowledge related to the corresponding semantic key is contained in the matrix. The embedded knowledge can be related to the co-occurred words or relevant entities in the training data of external memory. Some examples are shown in Table VII.

TABLE VII: Generated Responses of Knowledge Introduction by External Memory

Post	你最喜欢那个星座? What is your favorite constellation?	
Semantic-Key	Key-Word	Response
	摩羯座 Capricorn	我最喜欢摩羯座。 I love Capricorn.
	射手座 Sagittarius	我是射手座,我喜欢。 I am a Sagittarius and I like it.
Post	今天周杰伦的电影,太好看了。 Jay Chou's movie is so nice today.	
Semantic-Key	Key-Word	Response
	歌 song	杰伦的歌也很好听。 Jay Chou's songs are nice too.
Post	梵高的画,很漂亮。 Van Gogh's paintings are very beautiful.	
Semantic-Key	Key-Word	Response
	向日葵 sunflower	梵高笔下的向日葵。 The Sunflower by Van Gogh.
	荷兰 Holland	梵高的画,很漂亮。 Van Gogh's paintings are very beautiful.

As can be observed from the table, external information can be introduced by manually assigning specific semantic keys, which makes the comments substantial without hurting fluency and coherency. For example, given the semantic key "Song", the model shows the association ability and generated the comments "Jay Chou's songs are nice too.", which uses the knowledge that Jay Chou is a movie star and singer. However, it is not possible to always successfully introduce the knowledge into the comments when the specific semantic keys do not contain enough information. For example, given the semantic key "Holland", the generated comment is same as the post, which does not use the knowledge that Van Gogh is a painter of Holland. This is because that the content of the semantic key does not contain such knowledge.

V. Conclusion

This paper proposes a new generation approach for *short text conversation* task. By incorporating external semantic memory in encoder-decoder framework, the approach greatly alleviates the problem of general replies without substantiality and generates more diverse and concrete responses. Both objective evaluation and human evaluation demonstrate the advantages of this new approach. The separation of external memory construction and neural network training also makes it possible to utilize non-parallel corpora. Furthermore, the semantics of generated responses can be controlled by manipulating the semantic key mapper, which implies a new way to generate rich responses. Due to data sparsity, when data driven mapping functions like embedding or random mapping are used, the systems may generate incomprehensible comments, which is a problem to be addressed in the future.

Acknowledgement

We thank the STC challenge organization committee to provide human labelled STC corpora.

References

[1] Makoto P Kato, Kazuaki Kishida, Noriko Kando, Tetsuya Saka, and Mark Sanderson, "Report on ntcir-12: The twelfth round of nii testbeds and community for information access research," in *ACM SIGIR Forum*. ACM, 2017, vol. 50, pp. 18–27.

[2] Zongcheng Ji, Zhengdong Lu, and Hang Li, "An information retrieval approach to short text conversation," *arXiv preprint arXiv:1408.6988*, 2014.

[3] Lifeng Shang, Zhengdong Lu, and Hang Li, "Neural responding machine for short-text conversation," *arXiv preprint arXiv:1503.02364*, 2015.

[4] Oriol Vinyals and Quoc Le, "A neural conversational model," *arXiv preprint arXiv:1506.05869*, 2015.

[5] Ilya Sutskever, Oriol Vinyals, and Quoc V Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[8] Alexander M Rush, Sumit Chopra, and Jason Weston, "A neural attention model for abstractive sentence summarization," *arXiv preprint arXiv:1509.00685*, 2015.

[9] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko, "Sequence to sequence-video to text," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4534–4542.

[10] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan, "A diversity-promoting objective function for neural conversation models," *arXiv preprint arXiv:1510.03055*, 2015.

[11] Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan, "A neural network approach to context-sensitive generation of conversational responses," *arXiv preprint arXiv:1506.06714*, 2015.

[12] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[13] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[14] Minh-Thang Luong, Hieu Pham, and Christopher D Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[15] Jun Yin, Xin Jiang, Zhengdong Lu, Lifeng Shang, Hang Li, and Xiaoming Li, "Neural generative question answering," *arXiv preprint arXiv:1512.01337*, 2015.

[16] Yuanlong Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil, "Generating high-quality and informative conversation responses with sequence-to-sequence models," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2210–2219.

[17] Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma, "Topic aware neural response generation.," in *AAAI*, 2017, pp. 3351–3357.

[18] Yiping Song, Rui Yan, Xiang Li, Dongyan Zhao, and Ming Zhang, "Two are better than one: An ensemble of retrieval- and generation-based dialog systems," *arXiv preprint arXiv:1610.07149*, 2016.

[19] Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley, "A knowledge-grounded neural conversation model," *arXiv preprint arXiv:1702.01932*, 2017.

[20] Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu, "Emotional chatting machine: Emotional conversation generation with internal and external memory," *arXiv preprint arXiv:1704.01074*, 2017.

[21] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing, "Controllable text generation," *arXiv preprint arXiv:1703.00955*, 2017.

[22] Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin, "Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation," *arXiv preprint arXiv:1607.00970*, 2016.

[23] Lili Yao, Yaoyuan Zhang, Yansong Feng, Dongyan Zhao, and Rui Yan, "Towards implicit content-introducing for generative short-text conversation systems," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2190–2199.

[24] Iulian Vlad Serban, Tim Klinger, Gerald Tesauro, Kartik Talamadupula, Bowen Zhou, Yoshua Bengio, and Aaron C Courville, "Multiresolution recurrent neural networks: An application to dialogue response generation.," in *AAAI*, 2017, pp. 3288–3294.

[25] Wanxiang Che, Zhenghua Li, and Ting Liu, "Ltp: A chinese language technology platform.," *Journal of Chinese Information Processing*, vol. 2, no. 6, pp. 13–16, 2010.

[26] Alex Graves, Greg Wayne, and Ivo Danihelka, "Neural Turing Machines," *Arxiv*, pp. 1–26, 2014.

[27] Baolin Peng, Kaisheng Yao, Li Jing, and Kam Fai Wong, "Recurrent neural networks with external memory for spoken language understanding," *Eprint Arxiv*, vol. 9362, pp. 25–35, 2015.

[28] Per Christian Hansen, "The truncated svd as a method for regularization," *BIT Numerical Mathematics*, vol. 27, no. 4, pp. 534–553, 1987.

[29] Thomas K Landauer, *Latent semantic analysis*, Wiley Online Library, 2006.

[30] John A Hartigan and Manchek A Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[31] Lifeng Shang, Tetsuya Sakai, Zhengdong Lu, Hang Li, Ryuichiro Higashinaka, and Yusuke Miyao, "Overview of the ntcir-12 short text conversation task.," in *NTCIR*, 2016.

[32] Lifeng Shang, Tetsuya Sakai, Hang Li, Ryuichiro Higashinaka, Yusuke Miyao, Yuki Arase, and Masako Nomoto, "Overview of the NTCIR-13 short text conversation task," in *Proceedings of NTCIR-13*, 2017.

[33] Diederik Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[34] David M Blei, Andrew Y Ng, and Michael I Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[35] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng, "A biterm topic model for short texts," in *Proceedings of the 22nd international conference on World Wide Web*. ACM, 2013, pp. 1445–1456.

[36] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002, pp. 311–318.

[37] Chia-Wei Liu, Ryan Lowe, Iulian V Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau, "How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation," *arXiv preprint arXiv:1603.08023*, 2016.

[38] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.