CKIP at the NTCIR-14 STC-3 Task

Yi-Lin Xie[1] and Wei-Yun Ma[2]

[1] National Tsing Hua University, Taiwan
[2] Academia Sinica, Taiwan
yee0@nlplab.cc, ma@iis.sinica.edu.tw

Abstract. We describe how we build the system for NTCIR-14 Short Text Conversation (STC-3) Chinese Emotional Conversation Generation (CECG) subtask. In our approach, we first build an emotion keyword list, which contains emotion vocabularies under 5 different emotion classes. After that, responses are generated through a RNN-based inference mechanism, aimed at using emotion words from previously built keyword list to express emotion. Although our system doesn't reach top performance, we still find that there are some interesting findings in our results worth discussing.

Team Name. CKIP

Subtasks.
CECG

Keywords: artificial intelligence · dialogue systems · sequence-to-sequence · encoder-decoder · deep learning · recurrent neural network · long-short-term-memory · natural language processing · short text conversation · emotional conversation generation

1 Introduction

Human-Computer Interaction is a hot research topic in natural language processing (NLP). Due to the rise in the availability of social media and mobile devices, thousands of digital contents are created on the Internet everyday, especially short text messages. Commonly posted on the microblogging services such as Twitter and Weibo, short text message plays an important role in how we communicate with other people, making Short Text Conversation (STC), one round of conversation that consists of two short texts, a significant part among all the most common NLP tasks.

In recent years, deep learning approaches have advanced sentence generation significantly. Sequence-to-sequence generation framework [1] with attention mechanism [2][3] has gained great success in Machine Translation. These approaches were soon applied to a variety of other tasks, including Text Summarization, Grammatical Error Correction, Image Captioning, and Dialogue Modeling. As for STC, Neural Responding Machine [4] formalizes the generation of response as a decoding process based on the latent representation of the input text,

proven to better generate grammatically correct and content-wise appropriate responses. Moreover, different from the mainly STC works, seq2BF [5] employs a backward and forward decoding process to have responses contain additional topic words, which achieves controllable generation and improves content quality and diversity. In general, such approach's results show that the performance outperforms the traditional retrieval-based and SMT-based methods.

Even though much work has done on attaining response coherence and fluency, it is still a challenge to have machine generate responses with adequate emotion. The NTCIR-14 STC-3 Chinese Emotional Conversation Generation (CECG)[6], contrary to the STC task in previous year, specifically focuses on solving aforementioned problem. Inspired by Emotional Chatting Machine [7], the task aims to give appropriate responses to a given query with an additionally specified emotion category. Provided with Weibo post-comment dataset by the organizer, challengers have to come up with a proper way to train their models by leveraging these resources, making machine able to generate replies not only informative but also emotional.

In this paper, we present an inference approach based on the idea of seq2BF [5], enabling responses to contain emotion words, which is designed to match the user-specified emotion categories. The rest of the article is organized as follows. We describe our methodology in the next section. Then we describe the experiment settings in section 3. We report our system performance and discuss the evaluation results in section 4. Finally, we conclude our paper and explore the future direction in section 5.

2 Methodology

In this section, we describe our methodology in detail. To summarize, our approach consists of two steps:

1. Build an emotion keyword list.
2. Apply seq2BF inference process, using specific words to express emotion.

2.1 Emotion keyword list

Unlike generic STC task, generating emotional responses, as described in CECG overview [6], requires a chatbot to understand how to express the assigned emotions and affections. A simple yet efficient way is to make sentences contain emotion words, which have such affections consistent with the specified emotion classes. With this in mind, we thus build an emotion keyword list, aiming at helping machine use specific word to express emotion. Emotion keyword list contains emotion words under 5 different emotion classes, including Like, Sadness, Disgust, Anger, Happiness (see Table. 1).

2.2 Emotional Sentence Generation

Here, we introduce our sentence generation strategy based on the sequence-to-sequence framework (seq2seq, or encoder-decoder architecture) [1], where both

Table 1. Emotion keyword list

Emotion	**keywords**
Like	喜欢, 欣赏, 爱, 喜爱, 珍惜
Sadness	伤心, 难过, 悲伤, 心酸, 悲哀
Disgust	厌恶, 厌烦, 讨厌, 嫌弃, 反感
Anger	生气, 发火, 发怒, 恼怒, 气愤
Happiness	开心, 高兴, 快乐, 欢乐, 快活

encoder and decoder are recurrent neural networks (RNNs). Typical NMT models are based on this framework with attention mechanism [2]. The encoder first maps a source sentence $q = (q_1, q_2, ..., q_m)$ to a set of continuous representations $z = z_1, z_2, ..., z_m$. Given z, the decoder then generates a target sentence $r = (r_1, r_2, ..., r_n)$ of word tokens one by one. At each decoding step t of model training, the probability of generating a token r_t is maximized conditioned on q and $r_{<t} = (r_1, r_2, ..., r_{t-1})$. Given N training sentence pairs $\{q^i, r^i\}_{i=1}^N$, maximum likelihood estimation (MLE) is usually adopted to optimize the model, and the training objective is defined as:

$$L_{MLE} = \sum_{i=1}^{N} \log p(r^i|q^i) = \sum_{i=1}^{N} \sum_{t=1}^{n} \log p(r_t^i|r_1^i, ..., r_{t-1}^i, q^i), \tag{1}$$

where n is the length of sequence r^i.

Despite the good performance on frequently observed STC tasks, training seq2seq models with MLE, however, may get results lacking emotion words, which is determined by the word distribution of training data. To address this issue, we design a generation mechanism modified from the sequence-to-backward-and-forward-sequences model (seq2BF) [5] with an emotional-keyword injection mechanism to achieve the goal. Our model comprises a forward and a backward seq2seq model, where the first one is trained with standard way, and the second one is trained using responses with reversed word order. For example, given a training pair <"你想吃什么?(What do you want to eat ?), "我想吃面。(I want to eat noodles .)">, the forward sequence model is trained to answer "我想吃面。(I want to eat noodles .)" to the query "你想吃什么?", whereas the backward sequence model is trained to answer "。面想吃我(. noodles eat to want I)".

Once the both models are built, an emotional-keyword injection mechanism is applied during the inference phase, which is introduced as follows, (see Fig. 1). Given a query and a keyword to be inserted, the forward sequence generator first encodes the query and decodes a "half" reply $(r_s, r_{s+1}, ..., r_{n-1}, r_n)$ (see Step 1 in Fig. 1). In this step, we feed the given keyword as input at the beginning steps of decoding, and ignore the generator's output corresponding to it. A keyword is indicated as $(r_s, ..., r_{s+k-1})$, where k is the length of keyword (red words

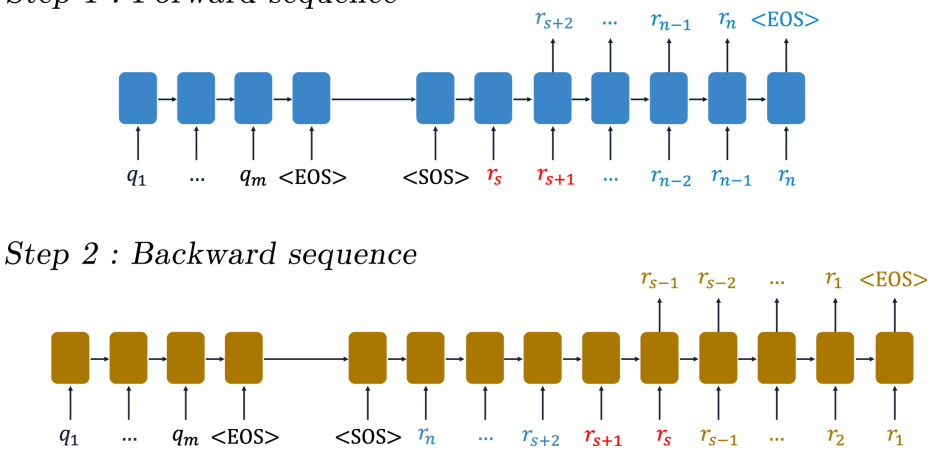

Fig. 1. An overview of our content-introducing approach to generate keyword-contained sentences. The forward sequence is first generated given the keyword to be inserted. $(r_s, ..., r_{s+k-1})$ is the keyword specified in advance as the constraint, whose length is indicated by k (set to 2 in this figure). The remainder is then completed by the backward sequence generator. Finally, we get the hypothesis by reverse the output in Step 2, i.e. $(r_1, r_2, ..., r_n)$.

in Fig.1) [3]. The backward sequence generator then encodes the query again, but decodes the entire reply, provided that the forward half of the reply with reversed word order is given (see Step 2 in Fig. 1). Finally, we reverse the output of backward sequence generator as the response, i.e. $(r_1, r_2, ..., r_n)$.

In practice, to generate a response given a query and an emotion category, we take all emotion words under the specified emotion class in emotion keyword list as the input of the model. Those whose sentence length exceeds 50 words will be filtered first, as too long sentences might be ungrammatical. After that, we choose the longest hypothesis as the answer, which we consider the most informative response.

3 Experiments

3.1 Data Preparation

The training dataset we use is built from more than 1 million Weibo posts and replies/comments, mixed with 600,000 pairs generated in this year, and 1,119,207 released in challenge 2017.

We preprocess the training data in the following ways: 1) For Weibo posts and replies/comments, sentences are usually expressed in short text messages

[3] Character-based models are applied in this work. Sometimes a keyword could be longer than one character.

without punctuations, making the context less clear. Therefore, we redo sentence tokenization by replacing spaces with commas. For example, "遇到问题　不要想着如何解释　如何推卸　要想如何去解决" (When having a problem, don't think of how to explain it or how to shirk it. Think of how to solve it.) will be processed into "遇到问题，不要想着如何解释，如何推卸，要想如何去解决". 2) We limit the number of consecutively repeated tokens (including punctuations) to at most three by truncating. For example, "干得好！！！！！" (Good Job!!!!!) will be truncated to "干得好！！！" (Good Job!!!).

3.2 Model and Training Details

We build our forward and backward seq2seq models upon the global attention mechanism [3], and use Long Short-Term Memory (LSTM) for information processing [8]. More precisely, the encoder is a 2-layer bidirectional LSTM, and the decoder is a 2-layer LSTM. The hidden layer size of both encoder and decoder is set to 256 and the word embedding size is set to 300. We apply dropout on both input tokens and embeddings with dropout rate set to 0.3.

Here, word representation and generation are performed on the character level, which allows us not only to mitigate unknown word issue, but also to reduce the computational time effectively. The vocabulary are shared between the source/target side, comprising 7K Chinese characters from the parallel data. The character embeddings are initialized with the pre-trained vector released by [9] [4], which is also trained with Weibo corpus. Embedding fine-tune is enabled until the training loss doesn't decrease.

We train our models by teacher forcing [10] and gradient clipping, used to address slow convergence of RNNs and prevent gradient explosion, respectively. We optimize our models using ADAM optimizer, with batch size 16 and learning rate 0.001. Each model is trained for 3 epochs on NVIDIA 1080 Ti GPU within one day.

4 Evaluation and Discussions

In this task, 1000 responses were evaluated by manual evaluation [6]. The labeling procedure is shown in the pseudo code described in Algorithm 1 and the score are computed by the formula listed below:

$$OverallScore = \sum_{i=0}^{2} i * num_i \tag{2}$$

$$AverageScore = \frac{1}{N_t} \sum_{i=0}^{2} i * num_i \tag{3}$$

[4] https://github.com/Embedding/Chinese-Word-Vectors

Algorithm 1 Labeling procedure

```
if (Coherence and Fluency) then
    if (Emotion Consistency) then
        LABEL 2 ## Score 2 for perfect responses
    else
        LABEL 1 ## Score 1 for coherent and fluent
    end if
else
    LABEL 0 ## Score 0 for others
end if
```

where num_i is the number of pairs which has a label of i for each submission run, and N_t is the total number of pairs for each run.

We made two submissions as follows:

- **CKIP_1** is done only through forward sequence generation (only Step 1 in Fig. 1), where the hypothesis we get is $(r_s, ..., r_n)$.
- **CKIP_2** is done using both forward and backward sequence generator, as the same way described in section 2.2.

Table 2 shows a case as a comparison to give an intuitive understanding of what is the difference between these two runs. As shown in this case, some of the responses generated by CKIP_2 has more words at the beginning of sequence than those generated by CKIP_1. Such these words are generated through the backward sequence generation, which step CKIP_1 is devoid of (see red words in Table 2).

4.1 Results

Table 3 shows the evaluation results of our two submitted runs.

According to the evaluation results [6], each of our runs doesn't get into top 80 out of all runs in this contest. Based on the observation, most of our generated responses still lack fluency and coherence. We conjecture that the main reason is due to exposure bias problem, which we will discuss in section 4.2.

Despite the large amount of responses labelled as 0, the number of our Label-2 responses is acceptable while the one in Label-1 is only a few. In other words, if a response is coherence and fluent, it is very likely to be emotion consistent too, which is identical to our secondary objective (i.e. to hit the given emotion). To verify this assumption, we re-examine the results by evaluating with the metric below:

$$EmotionConsistencyScore = \frac{\#Label\ 2}{\#Label\ 1} \tag{4}$$

where $\#Label\ 1$ and $\#Label\ 2$ is the number of pairs having a label of 1 and 2.

After applying this evaluation metric, our results outperforms all the other runs, showing the full potential of this idea. Consequently, we can actually say that our approach performs reasonably well in hitting the emotions we give it.

Table 2. Case study

Post	**最近大爱adele的歌** **Recently, I'm so addicted to adele's songs**	
Submission	**Emotion**	**Response**
CKIP_1	None	我也爱听 I love to listen too
	Like	喜欢shing的歌 Like shing's songs
	Sadness	悲哀的是我们都没有听过 Sad is all of us haven't listened it yet
	Disgust	讨厌，我也爱你 Yuck, I love you too
	Anger	恼怒的是我们的歌声是我的歌 Angry is that our voice is my song
	Happiness	开心的歌 Happy song
CKIP_2	None	我也爱听 I love to listen too
	Like	我喜欢shing的歌 I like shing's songs
	Sadness	最悲哀的是我们都没有听过 Saddest is all of us haven't listened it yet
	Disgust	讨厌，我也爱你 Yuck, I love you too
	Anger	最恼怒的是我们的歌声是我的歌 Angriest is that our voice is my song
	Happiness	这么开心的歌 So happy song

Table 3. Comparison between two submitted runs from CKIP

Submission	**Label 0**	**Label 1**	**Label 2**	**Total**	**Overall score**	**Average score**
CKIP_1	845	29	126	1000	281	0.281
CKIP_2	840	28	132	1000	292	0.292

4.2 Analysis

In this part, we further discuss the reason why most of our results lack coherence and fluency. We believe such an issue is caused due to the exposure bias: since each model has never explored the specified emotion words during training phase, it results in the discrepancy between training and inference stage. With this in mind, we re-examined our submitted results, finding that much cases is responded well if an emotion is not given, that is, generation is done via a standard seq2seq model. Here, we show one of the results suffering from this problem in Table 4.

In future work, we will further explore a different keyword giving strategy to handle this issue. For example, we can build the emotion keyword list using

Table 4. A result suffering from exposure bias from CKIP_2

Post	**为什么为什么为什么你们都不陪我看电影!！[怒骂]** **Why why why don't you accompany me to watch movie!! [curse]**
Emotion	**Response**
None	因为我们没有看到你 Because we didn't see you
Like	我很欣赏你的电影 I really appreciate your movie
Sadness	是啊，难过的是我们的电影院，我们都是一个人 Yeah, sad is our theatre, we are all alone
Disgust	我很厌烦的事情，我们都不想看电影，我们都不想看了 The thing I'm so disgusted, we don't want to watch movie, we don't want to watch
Anger	恼怒的是我们的电影院的电影院，我们都看了 Angry is the theatre of our theatre, all of us have watched
Happiness	很高兴的事情，我们一起看吧 Very happy thing, let's watch it together

the most frequent emotion words from the training data, or using an additional keyword classifier to specify emotion words according to the given query; both of them aim at reducing the irrelevance of the dialog.

5 Conclusions

In this paper, we describe our main approach that we use a forward and a backward sequence generation model to express emotion by containing emotion words. Instead of applying traditional seq2seq trained by MLE which might not be able to generate a sentence containing emotion words, we use an emotion keyword list and an inference strategy derived from seq2BF to express emotion. Despite the inability to get high score in official evaluation, our approach still show the potential of this idea through its capability to hit the emotion given. In the future, we will further explore a different keyword giving strategy to improve the performance, such as using most frequent emotion words from training data to build the emotion keyword list, and training a keyword classifier to specify emotion words.

References

1. Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.
2. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
3. Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

4. Lifeng Shang, Zhengdong Lu, and Hang Li. Neural responding machine for short-text conversation. *arXiv preprint arXiv:1503.02364*, 2015.
5. 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.
6. Yaoqin Zhang and Minlie Huang. Overview of NTCIR-14 short text generation subtask: Emotion generation challenge. In *Proceedings of the 14th NTCIR Conference*, 2019.
7. Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
8. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
9. Zhe Zhao, Tao Liu, Shen Li, Bofang Li, and Xiaoyong Du. Ngram2vec: Learning improved word representations from ngram co-occurrence statistics. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 244–253, 2017.
10. Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.