BUPTTeam at the NTCIR-13 STC-2 Task

Songbo Han
Beijing University of Posts and Telecommunications, China
hansongbo@bupt.edu.cn

Hao Zhu
Beijing University of Posts and Telecommunications, China
hzhu@bupt.edu.cn

Abstract

This paper provides an overview of BUPTTeam's system which participated in the NTICR-13 STC-2 task. STC-2 is a NTCIR challenging task which is defined as an information retrieval (IR) or natural language generation problem. In this paper, we propose a novel method to generate appropriate comments based on the following four steps: 1) preprocessing, 2) model building, 3) candidate comments generation, 4) candidate comments ranking. The evaluation results show that our methods finish the task successfully and have positive effect on improving the evaluation measurement.

Team name

BUPTTeam

Subtasks

Short Text Conversation (Chinese)

Keywords

Generation, Beam Search, Random Walk

1. Introduction

STC-2 is a NTCIR core task in NICIR-13, which aims at building a short text conversation (STC) system. This system can generate new comments for each post based on the provided post-comment repository (Shang et al., 2017).

Compared with NTCIR-12 STC task, NTCIR-13 STC considers not only retrieval-based method but also generation-based method. For the retrieval-based method, the basic idea is maintaining a large repository of short text conversation data (i.e. post-comment pairs), and finding a clever way to retrieve related comments from the repository and return the most appropriate one (Shang et al., 2017). For the generation-based method, it can build a suitable model based on the large repository so that it can generate more fluent and reasonable new comments for each post. And it is also helpful for researchers to find new approaches to build human-computer conversation.

Generation-based methods for STC fall into two categories, 1) the statistical machine translation(SMT) method and 2) the RNN-based (Recurrent Neural Network-based) neural models (Shang et al., 2017). RNN-based neural models are neural network models, which no longer needs artificial features. Besides, it can map the source language into target language directly.

There are four steps including preprocessing, building the model, candidate comments generation and candidate comments ranking in our method of generating new comments. And in this paper, we show the effectiveness of this method in generating short texts.

Our contribution is twofold: 1) we propose a novel method to make the generated sentences more fluent by applying n-gram to Seq2Seq model (Bahdanau et al., 2014). 2) we put forward a new way to construct graph for random walk algorithm to rank candidate comments.

2. System Architecture

The architecture of our STC system includes the following four components. As shown in Figure 1.

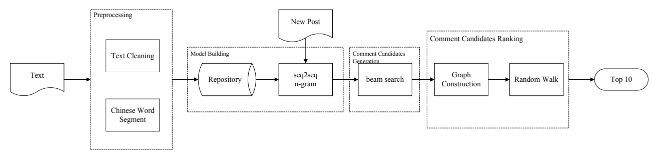

Figure 1: System Architecture

2.1 Preprocessing

In the original Weibo dataset, there are many special symbols, extra punctuation, and traditional Chinese characters. So, we need to clean the data in the first place. For example, we shall convert the traditional Chinese characters into simplified Chinese characters, remove the extra punctuation and special symbols.

As there is no obvious mark between chinses text word. Therefore, Chinese word segmentation is very important for text analysis. We use Stanford Word Segment[1] to split Chinese text in the Weibo dataset.

2.2 Model Building

The training data is crawled from the Sina WeiBo. It is rich in content and varied in style, so the sentence generated by the traditional Seq2Seq model is not fluent. To solve the problem, we propose a novel model to make the generated sentences more fluent by applying n-gram to Seq2Seq model. The model is described as Figure 2.

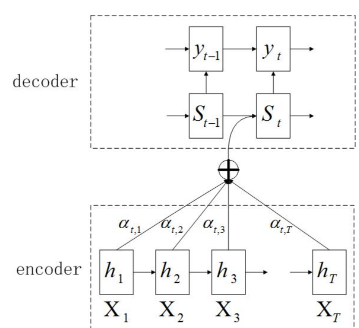

Figure 2: Model Architecture

In the model, an encoder maps the input sentence $\mathrm{x} = (\mathrm{x}_1, \ldots, \mathrm{x}_{\mathrm{T_x}})$ into the semantic vector c.

$$h_t = \mathrm{f}(x_t, h_{t-1}) \tag{1}$$

$$c = \mathrm{q}(\{h_1, \ldots, h_{T_x}\}) \tag{2}$$

[1] http://nlp.stanford.edu/software/segmenter.html

where $h_t \in R^n$ is a hidden state at time *t*. System uses a Long short-time memory(LSTM) neural network as f and $q(\{h_1, \dots, h_{T_x}\}) = h_{T_x}$ (Bahdanau et al., 2014).

The decoder aims to predict the next word y_t based on the semantic vector c and all the words $\{y_1, \dots, y_{t-1}\}$ that have been generated. In other words, the decoder can approximate a conditional probability distribution about y_i.

$$P(y_i|\{y_1, \dots, y_{i-1}, x) = k(y_{i-1}, y_i) * g(y_{i-1}, s_i, c_i) \quad (3)$$

where *x* is the input sentence, g is a nonlinear function, s_i is the hidden state of the LSTM (Bahdanau et al., 2014). The k function introduces 2-gram, which can make the generated result more fluent while maintain the generation capability of the model.

$$k(w_i, w_j) = \frac{count_{w_j}}{count_{w_i w_j}} \quad (4)$$

where $count_{w_j}$ is the frequency count of word w_j in the repository and $count_{w_i w_j}$ is the frequency count of phrase $w_i w_j$ in the repository.

The context vector c_i is calculated as a weighted sum of these annotations h_i (Bahdanau et al., 2014).

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \quad (5)$$

where h_i represent input sentence semantic focusing on the i-th word parts of the input sentence (Bahdanau et al., 2014).

The weight α_{ij} can be define as (Bahdanau et al., 2014):

$$\alpha_{ij} = \frac{e^{e_{ij}}}{\sum_{k=1}^{T_x} e^{e_{ik}}} \quad (6)$$

where

$$e_{ij} = \alpha(s_{i-1}, h_j) \quad (7)$$

system use a feedforward neural network as α function.

2.3 Candidate Comments Generation

For a given new post, the system generates candidate comments using the Beam Search algorithm. The Beam Search algorithm is a heuristic algorithm based on the branch and bound method. It heuristically estimates the k best paths, and only searches down from the estimated k paths. Only satisfied nodes will be retained in each layer while other nodes were permanently abandoned, In this way, the running time will be reduced greatly.

The Beam Search algorithm is applied to generate the comment according to the following three steps. Here, we set beam size as *k*.

1) Generate the probability of the word at the time step *t*.

2) Calculate the probability of the generated sentences from the time step 1 to *t*, and the system chooses the last word in the top *k* maximum probability sentences as the input of the time step *t* + 1.

3) Repeat step 1.

2.4 Candidate Comments Ranking

Instead of directly getting the result based on the probability of candidate comment for a new post, the system ranks the candidate comments using a graph-based random walk algorithm which can be divided into two part.

• Referent Graph Construction

Referent graph is a strongly connected graph represented by G= (V, E), where V is the set of all nodes and E is the set of all edges (Han et al., 2011).

Given a new post, the system generates 20 candidate comments by Beam Search algorithm and constructs a referent graph G. The graph G contains 20 nodes and edges, where each node represents a candidate comment. And there are two types of edges, where one is formed by comments and the other is formed by comment and post.

The weight of the edge formed by one candidate comment c_i and another c_j represent the semantic similarity $SR(c_i, c_j)$ (Han et al., 2011) which can be defined as:

$$SR(c_i, c_j) = \frac{v(i).v(j)}{||(v(i)||*||v(j)||} \quad (8)$$

where $v(c_i)$ and $v(c_j)$ is semantic vector of c_i and c_j respectively, which can be calculated as follows：

$$v(c_i) = \frac{1}{m}\sum_{k=1}^{m} w_{ik} \quad (9)$$

$$v(c_j) = \frac{1}{n}\sum_{k=1}^{n} w_{jk} \quad (10)$$

where w_{ik}, w_{jk} is pre-trained word2vector embedding.

The weight of the edge formed by one candidate comment c_i and the post represent the probability of generating c_i based on post p, which can be defined as:

$$score(p, c_i) = P_{model}(c_i|p) \quad (11)$$

The transition probability from post p to comment c_i and the transition probability from comment c_i to comment c_j can be calculated as (Han et al., 2011):

$$P(p \to c_i) = \frac{score(p,c_i)}{\sum_{c_i \in N_p} score(p,c_i)} \quad (12)$$

$$P(c_i \to c_j) = \frac{SR(c_i,c_j)}{\sum_{c_j \in N_{c_i}} SR(c_i,c_j)} \quad (13)$$

where N_p represent the generated comments set of the post p and N_{c_i} represent the adjacent comments set of the comment c_i in the graph G (Han et al., 2011).

• Ranking

When the referent graph G is constructed, the system will run the random walk algorithm.

Formula (14) and (15) illustrate the process of random walk with restart algorithm (Han et al., 2011).

$$r^0 = \alpha \quad (14)$$

$$r^{t+1} = (1 - \lambda) \times T \times r^t + \lambda \times \alpha \quad (15)$$

where α represent the probability distribution of the initial state of the referent graph G, r^t represent the intermediate result of random walk with restart, T represent times of iteration, and λ represent a parameter. let $r^{t+1} = r^t$, then we can get eventual stationary r (Han et al., 2011).

$$r = \lambda(I - mT)^{-1}\alpha, m = 1 - \lambda \quad (16)$$

For a post *p*, *E(c)* represent the effectiveness measure of a candidate comment *c*, which can be define as (Han et al., 2011):

$$E(c) = score(p, c) \cdot r(c) \quad (17)$$

Finally, the system ranks candidate comments according to E(c) and puts top ten comments in the ranking list as the result.

3. Experiments

3.1 Data Set

There are 219174 Weibo posts and the corresponding 4305706 comments in the repository (Shang et al., 2017).

There are 769 query posts and each has nearly 15 comment candidates in the training data (Shang et al., 2017). There are 11,535 post-comment pairs with “suitable”, “neural”, and “unsuitable” labels. “Suitable” means that the comment is clearly a suitable comment to the post and “neutral” means that the comment can be a comment to the post in a specific scenario, while otherwise the comment shall be labeled as “unsuitable”.

There are100 posts in the test data (Shang et al., 2017). We are requested to submit up to five runs for the task. In each run, a ranking list of ten comments for each test query is required.

3.2 Evaluation Metrics

The evaluation metrics include nG@1, nERR@10 and P+ (SaKai et al., 2015).

nG@1 shows the quantity of generated comments. (SaKai et al., 2015).

nERR@10 shows the reasonability of the ranking list, which means the better comment should be in front of the list. (SaKai et al., 2015).

P+ shows the quantity of the top comment in ranking list (SaKai et al., 2015).

3.3 Experimental Results

Table 1 include the best four teams’ and our results which have been sorted by Mean nG@1, P+ and nERR@10.

Table 1: Part of Official STC results (Shang et al., 2017)

Run	nG @1	Run	P+	Run	nERR @10
SG01-C-G1	0.58 41	SG01-C-G1	0.65 80	SG01-C-G1	0.7130
splab-C-G4	0.50 80	splab-C-G4	0.60 80	splab-C-G4	0.6492
srcb-C-G2	0.40 80	srcb-C-G2	0.51 88	srcb-C-G2	0.5781
TUA1-C-G4	0.38 93	TUA1-C-G4	0.49 09	TUA1-C-G4	0.5277
BUPTTe am-C-G1	0.18 23	BUPTTe am-C-G1	0.27 55	BUPTTe am-C-G1	0.2746

Table 1 shows that our proposed methods are not ideal in the STC task. And there is still a lot of room for improvement in the nG@1 indicator.

Table 2 shows the performance of our different runs in the task.

1) G1 generates sentence by applying n-gram based on Seq2Seq model. And the Beam Search algorithm is applied to the comment generated. And the random walk with restart is used to rank the candidate comments.

2) G2 generate sentence based on traditional Seq2Seq model. And the Beam Search algorithm is applied to the comment generated. And the random walk with restart is used to rank the candidate comments.

Table 2: Comparison of Performance on 2 Runs (Shang et al., 2017)

Run name	nG@1	P+	nERR@10
BUPTTeam-C-G1	0.1823	0.2755	0.2746
BUPTTeam-C-G2	0.0933	0.1895	0.2001

As we can see from the table 2, G1, using n-gram based on Seq2Seq model, can improve the evaluation measurement significantly.

4. Conclusions

In this paper, we propose a novel method for STC-2 task of NTCIR-13.

We apply n-gram to improve the fluency of the generated results based on the Seq2Seq model, where the Beam Search algorithm is used to generate candidate comments and the random walk which is a graph-based method is used to rank candidate comments. The evaluation results show that our method finish the task successfully and have positive effect on improving the evaluation measurement.

5. References

[1] Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate[J]. Computer Science, 2014.

[2] Daisuke Ishikawa, ASURA: A Best-Answer Estimation System for NTCIR-8 CQA Pilot Task, 2010.

[3] Daisuke Ishikawa, Tetsuya Sakai, Noriko Kando. Overview of the NTCIR-8 Community QA Pilot Task: The Test Collection and the Task. NTCIR-8 Workshop Meeting, 2010.

[4] Kazuko Kuriyama. Best-Answer Selection Using a Machine Learning Tool at NTCIR-8 CQA Pilot Task, 2010.

[5] Karl Pearson. The Problem of the Random Walk. Nature, 1905, 268: 2113–2122.

[6] Leo Grady. Random walks for image segmentation. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2006, 28(11): 1768–1783.

[7] Lifeng Shang, Tetsuya Sakai, Zhengdong Lu, Hang Li, Ryuichiro Higashinaka, and Yusuke Miyao. 2016. Overview of the NTCIR-12 Short Text Conversation Task. In Proceedings of NTCIR-12. 473–484.

[8] Lifeng Shang, Tetsuya Sakai, Zhengdong Lu, Hang Li, Ryuichiro Higashinaka, Yusuke Miyao, Yuki Arase. 2017. Overview of the NTCIR-13 Short Text Conversation Task. In Proceedings of NTCIR-13.

[9] Pi-Chuan Chang, Michel Galley, and Christopher D. Manning. Optimizing Chinese word segmentation for machine translation performance, ACL, 2008.

[10] Tetsuya Sakai, Lifeng Shang, Zhengdong Lu, and Hang Li. Topic Set Size Design with the Evaluation Measures for Short Text Conversation, 2015.

[11] Xianpei Han, Le Sun and Jun Zhao. Collective entity linking in Web Text: a graph-based method. Proceedings of International Conference on Research & Development in Information Retrieval. Beijing, 2011.

[12] Xiaoqing Zheng, Hanyang Chen and Tianyu Xu. Deep Learning for Chinese Word Segmentation and POS Tagging. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 2013.

[13] Young-In Song, Jing Liu, Tetsuya Sakai, Xin-Jing Wang, Guwen Feng, Yunbo Cao, Hisami Suzuki and Chin-Yew Lin. Microsoft Research Asia with Redmond at the NTCIR-8 Community QA Pilot Task, 2010.

[14] Yongmei Tan, Mingda Wang and Songbo Han. BUPTTeam Participation in NTCIR-12 Short Text Conversation Task, 2016.

[15] Zhaochen Guo, Denilson Barbosa. Robust Entity Linking via Random Walks. Proc. CIKM, 2014.

[16] Han X, Sun L, Zhao J. Collective entity linking in web text: a graph-based method[C]//Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval. ACM, 2011: 765-774.