UT Dialogue System at NTCIR-12 STC

Shoetsu Sato
Graduate School of Information Science and Technology,
The University of Tokyo
shoetsu@tkl.iis.u-tokyo.ac.jp

Shonosuke Ishiwatari
Graduate School of Information Science and Technology,
The University of Tokyo
ishiwatari@tkl.iis.u-tokyo.ac.jp

Naoki Yoshinaga*
Institute of Industrial Science,
The University of Tokyo
ynaga@tkl.iis.u-tokyo.ac.jp

Masashi Toyoda
Institute of Industrial Science,
The University of Tokyo
toyoda@tkl.iis.u-tokyo.ac.jp

Masaru Kitsuregawa
Institute of Industrial Science,
The University of Tokyo
National Institute of Informatics
kitsure@tkl.iis.u-tokyo.ac.jp

ABSTRACT

This paper reports a dialogue system developed at the University of Tokyo for participation in NTCIR-12 on the short text conversation (STC) pilot task. We participated in the Japanese STC task on Twitter and built a system that selects plausible responses for an input post (tweet) from a given pool of tweets. Our system first selects a (small) set of tweets as response candidates from the pool of tweets by exploiting a kernel-based classifier. The classifier uses bag-of-words in an utterance and a response (candidate) as features. We then perform re-ranking of the chosen candidates in accordance with the perplexity given by Long Short-Term Memory-based Recurrent Neural Network (LSTM-RNN) to return a ranked list of plausible responses. In order to capture the diversity of domains (topics, wordings, writing styles, etc.) in chat dialogue, we train multiple LSTM-RNNs from subsets of utterance-response pairs that are obtained by clustering of distributed representations of the utterances, and use the LSTM-RNN that is trained from the utterance-response cluster whose centroid is the closest to the input tweet.

Team Name

sss

Subtasks

Short Text Conversation (Japanese)

Keywords

conversation, clustering, domain adaptation, word embedding, neural network language model

1. INTRODUCTION

In the Japanese task of the NTCIR-12 short text conversation (STC) pilot task, participants need to develop a system that takes an input tweet and extracts, from a pool of tweets (utterance-response pairs), a (short) list of tweets that are ranked according to their relative suitability as response to the input. The size of the pool of tweets is around one million, which consist of 500K utterance-response pairs.

To solve this task, Long Short-Term Memory-based Recurrent Neural Networks (LSTM-RNNs) is used to evaluate the suitability of each response in the pool as response to the input tweet. The key features of our system are two-folds:

Response pre-filtering Since LSTM-RNNs are slow to evaluate the entire responses in the pool, we utilize a classifier to select a tractable number of tweets as response candidates. The classifier based on polynomial kernel is trained with a large number of utterance-response pairs that are independently crawled from Twitter.

Domain-aware LSTM-RNNs In chat dialogue on Twitter, the diversity of domains (topics, wordings, writing styles etc.) is evident. We therefore train multiple domain-aware LSTM-RNNs to evaluate the suitability of each response candidate as response. We obtain domain-consistent subsets of utterance-response pairs by clustering and train one LSTM-RNN from each subset. The LSTM-RNN obtained from a subset of utterance-response pairs whose utterances are semantically closest to the input tweet is used to evaluate the suitability of each response tweet as response to the input tweet.

In what follows, we detail the architecture of our system and briefly summarize the experimental results.

2. SYSTEM ARCHITECTURE

Figure 1 depicts our dialogue system used for NTCIR-12 STC pilot task. The numbers in Figure 1 indicate Sections in the following explanations.

2.1 Domain-aware dialogue modeling

Topics, wordings and writing styles (or domains) vary substantially in chat dialogue, which makes it difficult to build a universal dialogue model that can handle various domains. Our dialogue system is inspired by Yamamoto and Sumita's work on domain adaptation for statistical machine translation [6]. They showed that domain-specific models trained on smaller domain-specific corpora performed better than a general model trained on a larger general-domain corpus.

*This work is done while the author concurrently served as a senior researcher at National Institute of Information and Communications Technology (NICT), Japan.

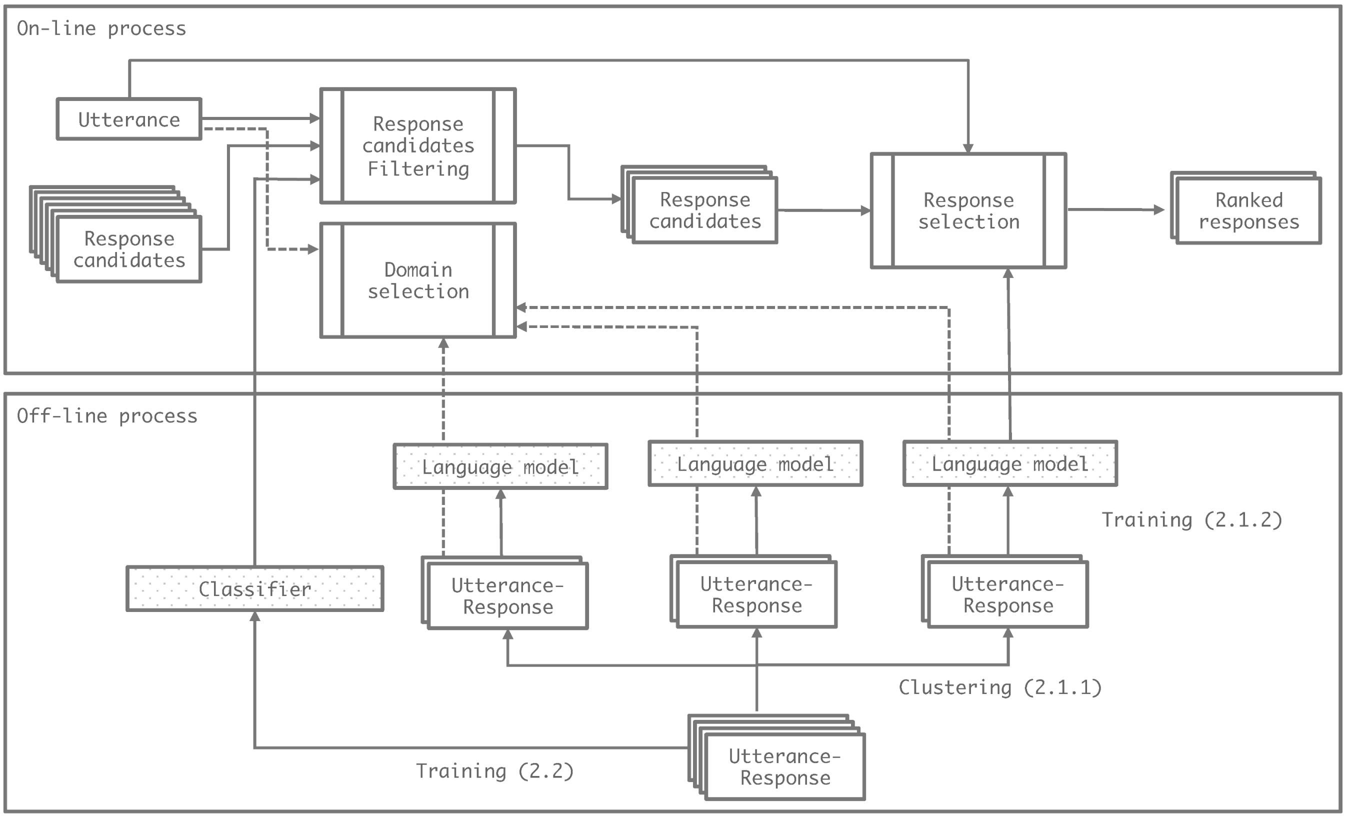

Figure 1: System overview.

The major challenge in adopting Yamamoto and Sumita's approach to model chat dialogue is in its domain diversity. To cover a variety of domains, we want to split data (for modeling dialogue) into pieces. However, as the amount of the data decreases, a data sparseness problem becomes more serious. Thus, there will be a trade-off between the performance improvement achieved by domain-specific data and the performance drop caused by data sparseness. Clarifying this trade-off is the main issue we focus on in this research.

2.1.1 Clustering vector representations of utterances

We first classify utterances in a given pool of utterance-response pairs (NTCIR tweets) into domain-consistent subsets by clustering. We represent the utterances in NTCIR tweets with distributed vector representations. These vector representations of utterances are obtained by averaging vector representations of words in the utterances.

The vector representations of words are induced from 2013 portion of our Twitter archive we have crawled since 2011 (UT tweets) in advance. The details of these tweets are summarized in Section 3.1. Since words in Japanese are concatenated and the text in tweets contains many out-of-dictionary words, we use MeCab[1] with mecab-ipadic-neologd[2] to tokenize the text. We then use word2vec skip-gram model [3] to induce vector representations of words from UT tweets.

Having vector representations for utterances in utterance-response pairs, we run k-means clustering to obtain clusters of utterance-response pairs. We regard each cluster as one domain, following Yamamoto and Sumita's work [6].

When choosing domain for input tweet, we obtain a vector representation of the tweet in the same way as above, and find the cluster (domain) whose centroid is the closest to the obtained vector representation in terms of the euclidean distance.

2.1.2 LSTM-RNNs for utterance-response model

We train LSTM-RNNs from clusters of utterance-response pairs obtained in Section 2.1.1. Each utterance and its response in the clusters is concatenated with a special symbol ([EOU]) to form a pseudo sentence. The resulting pseudo sentences in each cluster are given to an LSTM-RNN for training a domain-specific language model.

We should mention that RNNs encode temporal information implicitly for contexts with arbitrary length, which is proven to be more effective than classical n-gram models [2]. However, it is well known that a vanilla RNN suffers from the vanishing gradient problem. To overcome this problem, LSTM-RNNs are introduced and widely used in conversation modeling tasks [4].

The resulting LSTM-RNNs are used to evaluate how likely each response candidate in the pool of tweets is suitable as response to the input tweet. As in the training LSTM-RNNs, the input tweet and each response candidate is concatenated to form a pseudo sentence. A perplexity of the resulting pseudo sentence is used for the evaluation (to avoid the length factor of the response), and response candidates with lower perplexities are chosen as plausible responses to the input tweet.

[1] http://taku910.github.io/mecab/
[2] https://github.com/neologd/mecab-ipadic-neologd

2.2 Response candidate filtering

High computational cost regarding matrix multiplications in evaluating with LSTM-RNNs (Section 2.1.2) causes a practical issue when there exist hundreds of thousands of response candidates. We therefore incorporate a pre-filtering step to reduce the candidates before performing response selection using LSTM-RNNs. In this pre-filtering step, we choose the tractable number[3] of response candidates for the following LSTM-RNN evaluation according to the margins from the separating hyperplane that are provided by a margin-based binary classifier.

The classifier used as a filter is based on a variant of online passive-aggressive algorithm (PA-I) [1], and employs bag-of-words of an utterance-response pair as features. To capture combinations of words in utterance and response, we use a second-order polynomial kernel. Since the kernel evaluation is known to be slow, we adopt opal,[4] which implements fast kernel evaluation based on kernel slicing technique [7].

To train the classifier, we compile utterance-response pairs from (a part of) NTCIR tweets and UT tweets as positive examples, and prepare the same amount of negative examples by combining utterances with randomly-chosen responses.

3. EXPERIMENTS

This section evaluates our domain-aware dialogue system on the response retrieval task. To validate the effectiveness of domain-aware LSTM-RNNs, we first report experiments on a manually-built dataset, followed by results on the formal run of NTCIR-12.

3.1 Settings

We first built two sets of utterance-response pairs (tweets). We have crawled 421K (421,050) utterance-response pairs in 2014 (NTCIR tweets) from given 500K utterance-response ID pairs by using Twitter APIs. These are provided for NTCIR12 STC Japanese task.[5] In addition, we extracted 230M utterance-response pairs (UT tweets) in 2013 from Twitter archive we have been crawling since 2011.

We next induced vector representations for words in the utterances of the UT tweets by using skip-gram model (implfemented in word2vec[6]) with setting dimension to 200 and window size to 5. We then chose 100K utterance-response pairs from NTCIR tweets, and applied k-means clustering (implemented in scikit-learn toolkit[7]) to the vector representations of the utterances, as stated in Section 2.1.1.

We varied the number of clusters, k, from 1 to 40, and used the resulting clusters of utterances (and associated responses) to train LSTM-RNNs (implemented by TensorFlow[8]). The hyperparameters of LSTM-RNNs are tuned using a small set of utterance-response pairs taken from the NTCIR tweets.

To solely validate the effectiveness of domain-aware LSTM-RNNs and investigate the effect of the number of clusters, we manually built a small test set in the following way. We first sampled 1K utterance-response pairs from NTCIR tweets, and assumed the utterances as input tweets and the responses as correct (or appropriate) responses. We next extracted 19 responses for each input tweet from UT tweets and regard them as (additional) candidates (wrong responses). We then obtained 1K problems (1 input tweet and 20 response candidates including one correct one). We directly use the above LSTM-RNNs to evaluate the suitability as response.

On the other hand in the NTCIR-12 formal-run dataset, 204 input tweets in 2015 are given and the above 421K responses are regarded as response candidates. As stated in Section 2.2, a kernel-based classifier (implemented in opal[4]) is leveraged to select plausible candidates from the entire response candidates. We have used the NTCIR tweets (excluding 200 pairs for evaluating pre-filtering) augmented with randomly sampled UT tweets (in total, around 8.4M training examples) to train the classifier, and chose 500 candidates for each input tweet in accordance with a margin from the separating hyperplane of the classifier. We will also solely evaluate the effectiveness of the pre-filter later.

system	accuracy@3
random	15.0%
baseline ($k = 1$)	30.8%
proposed ($k = 10$)	33.2%
proposed ($k = 20$)	35.4%
proposed ($k = 40$)	35.0%

Table 1: Results on the small test set: accuracy@3 is the proportion of the input tweets where the top-3 response candidates chosen by the system included the correct response.

3.2 Results on the small test set

Table 1 lists experimental results on the small test set built in Section 3.1. We can clearly observe that our proposed systems with multiple LSTM-RNNs ($k > 1$) outperformed the system with a single LSTM-RNN ($k = 1$, baseline). The performance is saturated when $k = 20$, which indicates the trade-off between data sparseness and domain consistency.

We next investigated the detailed performance of the best-performing system ($k = 20$) and baseline ($k = 1$) on the input tweets in each cluster when $k = 20$ (Table 2). The proposed system ($k = 20$) outperformed baseline for 13 out of 20 clusters (and ties for 3 clusters). The improvement is evident for larger size of clusters (#elems (train) > 5000). Considering that the number of the utterance-response pairs for training LSTM-RNNs was reduced significantly from 100K (baseline) (0.7% (ID: 11) to 11.8% (ID: 13)), use of consistent subsets of the training data compensated for the reduction of the training data.

The performance drop against baseline is attributed to the increase of unknown words due to the data sparseness problem. This issue will be ameliorated by adopting a soft clustering method that allows us to interpolate all the LSTM-RNNs to evaluate the suitability of tweets as response.

Table 3 shows input tweets and the correct responses along with selected responses, in which our method returned the correct or more appropriate results than baseline system. The baseline system often returned wrong responses that frequently observed in training data (e.g., greetings domain or acknowledgments domain). Our system divided these common but harmful responses into other clusters so that it can obtain more consistent LSTM-RNNs to select correct responses.

[3] We choose 500 candidates in the experiments.
[4] http://www.tkl.iis.u-tokyo.ac.jp/~ynaga/opal/
[5] https://github.com/mynlp/stc
[6] https://code.google.com/archive/p/word2vec/
[7] http://scikit-learn.org/stable/
[8] https://www.tensorflow.org/

ID	domain (topics, wording, writing style)	#elems train	#elems test	#corr ours	#corr baseline	improvement $\frac{\Delta\#\text{corr}}{\#\text{elems (test)}}$
13	-	11801	108	**38**	27	10.19%
7	-	11524	124	**37**	32	4.03%
14	politics, economics, social matters	10294	130	**48**	38	7.69%
3	-	9743	94	**32**	23	9.57%
16	animation, comics	6747	56	**11**	10	1.79%
12	-	6552	66	**24**	23	1.52%
19	game	5677	50	**13**	5	16.00%
10	-	5627	45	**14**	13	2.22%
1	end with '?' r '!'	5190	63	**17**	15	3.17%
0	moaning (esp., sleepy, weary)	5064	52	17	**21**	-7.69%
15	-	4908	50	22	**24**	-4.00%
17	numbers	3803	31	5	**7**	-6.45%
6	eating	2630	16	**6**	4	12.50%
2	frank acknowledgment (follow, RT)	2252	33	29	**30**	-3.03%
18	end with '!!!'	1869	17	**8**	**8**	0.00%
8	polite acknowledgement (follow, RT)	1553	13	**12**	**12**	0.00%
4	greetings	1537	21	**7**	6	4.76%
9	end with '· · ·'	1326	12	**3**	2	8.33%
5	polite morning greetings	1174	13	**9**	6	23.08%
11	shouting with word lengthing or repetition	729	6	**2**	**2**	0.00%
total		100000	1000	354	308	4.60%

Table 2: Results on each cluster of the small test set: our system with $k = 20$ clusters vs. baseline ($k = 1$); #corr is the number of tweets in which the correct response was included in top-3 response candidates.

Utterance	Correct Response	baseline	proposed ($k = 20$)
あ、見るの忘れてた。おめでとう！	ありー！見直してくれてありがとう！	今年は1年ありがとうございました	ありー！見直してくれてありがとう！
映画も見ました？僕は両方見ました！	映画も見ましたよ—	ありがとうございます！よろしくお願いします！	映画も見ましたよ—
うーん今日は真面目に練習しようかしら……取り敢えずお昼いってきます	いてらですー。自分も食ってきます	司令官、おはよう。今日もよろしくね。	いてらですー。自分も食ってきます
一言のやつ相変わらずバカなんでしょうね	それだけかい！笑	おかえりなさい！お疲れ様でした	それな！楽しみにしてる!!
そちらの学校は楽しいの？	楽しいよ。	お疲れ様でした！	楽しいよ。
TL掘ったら鯱が海外進出だって!?北京？笑中国に行きたくねー笑	え、なにそのじょうほう	一年お疲れ様でした～。来年は◯◯さんのセカンドですね！	なんですか
ベジータが体モノマネと対になってたってことだよね。すごい…。	それね！ほんとすごいですよね！	おはようございます。家に帰るのが怖いです笑	それね！ほんとすごいですよね！
アプリのランキングにひどいやつがあって笑った	◯◯、16日てご飯食べいくよね？	うわ、お大事にしてください。	たまにありますね。あれ、なんなんでしょう。
俺も九段下の駅をおりて坂道を人の流れ追い越して行きてーなー	そこから見えるひかるたまねぎ見たことある??	え？ええっっ!?おめですぅぅ!!!	そうそう！結構邪魔よね、あれ。
すいません、◯◯さんのアカウントを勝手に使ってしまって本当にすいません…遊び半分で入れてみたら本当に信じるとは思っていなかったので…	全然いいですよむしろ嬉しかったですw 俺の名前を使ってくれたのが	おはよー！	全然いいですよむしろ嬉しかったですw 俺の名前を使ってくれたのが
質問です!! LINEで話したことない先輩をブロックして削除するのはやめた方がいいですか？	やめた方がいいと思うよ・・・	おつかれさまです・・・！	やめた方がいいと思うよ・・・
あれ、昨日飲みまくってなんかやらかしたかな六万くらい消えてるまぁいいか	6万飛ばすなんてなんて大人だ・・・	うわあああああ！なつかしい!	お疲れ様でした。良いライブになったかな？
カントリーマァムのドリンクのやつが見つかりません。	ローソン限定じゃなかったっけ？	先輩、おはよーございます♪	ローソン限定じゃなかったっけ？
ロックスターとレッドブルを連続で飲むと気持ち悪いね。特にね。	当たり前だよね。今日はね。ありがとうね。	来年もよろしくお願いいたします。	アヤちゃんだね。。癒されたよね。。

Table 3: Example input tweets for which our system returned better results than baseline (top-1 responses).

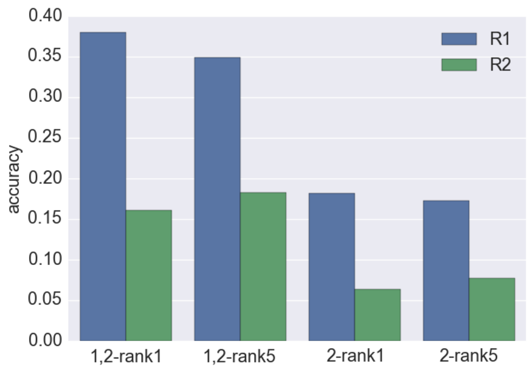

Figure 2: Evaluation results on formal run test.

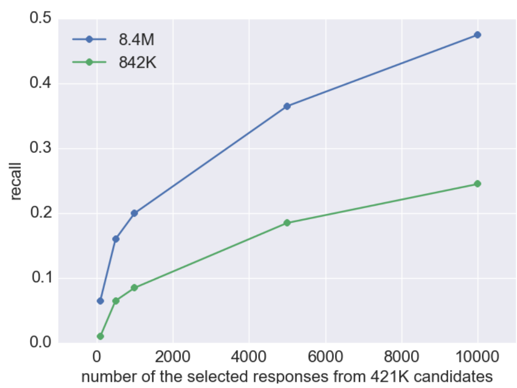

Figure 3: Filtering performance.

3.3 Results on NTCIR-12 formal-run datasets

The results on NTCIR-12 formal-run datasets were manually evaluated by multiple human annotators in the following way. The annotators are asked to examine the top-1 or top-5 response candidates returned by a system and to assign score of 0 (inappropriate), 1 (appropriate in some context), and 2 (appropriate) to each response.

We have provided two systems for this evaluation. One is the proposed system ($k = 20$) pre-filtered by the kernel-based classifier (R1), while the other just returns the top-10 pre-filtered responses (R2).

Figure 2 shows the results. 1,2-rank$n$ refers to the accuracy of the top-n responses assuming those scored with 1 or 2 as correct, while 2-rank$n$ refers to the accuracy of the top-n responses assuming those scored with 2 are correct. For all the cases the LSTM-RNNs improved the accuracy against the responses chosen by the kernel-based classifier.

To analyze the effectiveness of the filtering step, we evaluated the recall of two pre-filters (classifiers). We randomly sampled 200 utterances from NTCIR tweets as input tweet, and chose the top-N response candidates from 421k (421,050) responses in NTCIR tweets. To see the impact of the size of training data, another pre-filter is trained with 842K training examples created only from NTCIR tweets (421K (421,050) pairs) excluding 200 pairs used for the test set, in addition to the pre-filter (R1) trained with 8.4M examples.

Figure 3 shows the recall of the pre-filters plotted against the number of selected response candidates, N. Here, recall is the proportion of input tweets for which the top-N response candidates returned by pre-filters included the correct responses. In the formal-run, our LSTM-RNN model selected responses from filtered top-500 response candidates. Figure 3 shows that use of larger training data (8.4M) significantly improved the recall of top-500 responses from 6.5% to 16%. This is significantly higher than random sampling ($500/420850 = 0.001$) but is not high enough unless there are 5 ($\sim 100/16 - 1$) alternative responses other than the correct ones in the given pool of tweets.

We will increase the training data size of a pre-filter to improve the recall and accelerate the evaluation of LSTM-RNNs to increase the number of processible response candidates.

4. CONCLUSIONS

Our system for the Japanese task of NTCIR-12 short text conversation pilot task is presented. Our system has made LSTM-RNNs scalable for this task by choosing a (small) set of tweets as response candidates, from a large pool of tweets, using a kernel-based classifier. To capture the diversity of domains in chat dialogue, we have trained multiple LSTM-RNNs for consistent subsets of utterance-response pairs obtained by applying k-means clustering to their distributed representations. The effectiveness of the multiple LSTM-RNNs are validated through a manually-tailored testset, and they are successfully utilized at the formal run of NTCIR-12.

We are going to investigate the effectiveness of our method based on the recent sophisticated models like bi-directional LSTM [5] instead of LSTM-RNN we have employed here.

Acknowledgments

This work was partially supported by JSPS KAKENHI Grant Number 25280111, 16H02905, and 16K16109.

5. REFERENCES

[1] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

[2] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur. Recurrent neural network based language model. In *Proceedings of INTERSPEECH*, volume 2, page 3, 2010.

[3] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in NIPS*, pages 3111–3119, 2013.

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

[5] M. Sundermeyer, T. Alkhouli, J. Wuebker, and H. Ney. Translation modeling with bidirectional recurrent neural networks. In *Proceedings of EMNLP*, pages 14–25, 2014.

[6] H. Yamamoto and E. Sumita. Bilingual cluster based models for statistical machine translation. In *Proceedings of EMNLP-CoNLL*, pages 514–523, 2007.

[7] N. Yoshinaga and M. Kitsuregawa. Kernel slicing: Scalable online training with conjunctive features. In *Proceedings of COLING*, pages 1245–1253, 2010.