SG01 at the NTCIR-13 STC-2 Task

Haizhou Zhao[†] Yi Du[†] Hangyu Li[†] Qiao Qian[‡] Hao Zhou[‡] Minlie Huang[‡*] Jingfang Xu[†*]
[†]Sogou Inc., Beijing, China [‡]Tsinghua University, Beijing, China
{zhaohaizhou, duyi, lihangyu, xujingfang}@sogou-inc.com,
{qianq15, zhouhao16}@mails.tsinghua.edu.cn, aihuang@tsinghua.edu.cn

ABSTRACT

We describe how we build the system for NTCIR-13 Short Text Conversation (STC) Chinese subtask. In our system, we use the retrieval-based method and the generation-based method respectively. For the retrieval-based method, we develop several features to match the candidates and then apply a learning to rank algorithm to get properly ranked results. For the generation-based method, we first generate various high-quality comments and then do ranking to select better ones. As reported in the task overview, we have achieved top performance in both methods with three submissions for the retrieval-based method and five submissions for the generation-based method.

Team Name

SG01

Subtasks

Short Text Conversation (Chinese)

Keywords

Short Text Conversation, Information Retrieval, Learning to Rank, Neural Network, seq2seq, VAE.

1. INTRODUCTION

Developing bots to have natural conversations with humans is quite a challenging task. The Short Text Conversation task, first launched in NTCIR-12 as a pilot task NTCIR-12 STC [8], has defined a simplified version of bots conducting conversations with humans, and has provided a testbed for evaluating different approaches.

We participate in NTCIR-13 STC-2 [7] Chinese subtask. The task aims to give appropriate responses to a given query, either by a retrieval-based method or by a generation-based method. The retrieval-based method takes STC as an IR problem: given a repository with existing post-comment pairs, it reuses the comments to respond to queries. The generation-based method aims to generate new responses, which has been a hot research topic in recent years. To solve the STC task, we use both methods and have achieved top performance according to the task overview.

This paper is organized as follows: we will discuss our approaches for the retrieval-based method in Section 2 and then move on to the generation-based method in Section 3. In Section 4, we compare these approaches, wrap up and give the conclusions.

*Corresponding authors

2. RETRIEVAL-BASED METHOD

Our system performs retrieval-based STC in three stages. In the first stage named Retrieve Stage, we retrieve some post-comment pairs from the repository for each given query, mainly based on text similarity between the query and post-comment pairs in the given repository, forming a reduced candidate set. In the second stage named Ranking Stage I, we introduce more lexical and semantic features designed for STC scenario, then combine them to get a weighted score, by this, we can further reduce the candidate set. In the third stage named Ranking Stage II, we introduce more semantic features, together with all features calculated in preceding stages, we can use learning to rank to get top 10 comments as the final result. The philosophy we separate the process into stages is as follows: first we want to reduce the candidate set step by step, so we can postpone calculating complex features to later stages; second, we want to keep the candidate set effective, which means it should contain proper comments for the given query. Next, we describe the retrieval-based method in detail.

2.1 Data Pre-processing

We analyze the given repository and believe that some comments are not quite possible to be perfect candidates for any STC query. So we collect these comments and remove the related post-comment pairs from the given repository.

There are mainly three types of comments we want to remove from the repository:

- Comments with high frequency. They are always less informative.
- Comments with ad words. They are not likely to be replying any post.
- Comments less than or equal to two words after Chinese word segmentation.

2.2 Retrieve Stage

In this stage, we treat each post-comment pair as a virtual document, the post is like a title, and the comment is like the content of the document. We put the given repository into our IR system designed for web search task; for each test query, we fetch 500 candidate post-comment pairs from the given repository as a reduced candidate set $S_{reduced1}$. The selection criteria are based on a dozen of traditional features used in web search engine, such as BM25 [5], MRF for term dependency [4], Proximity [10], etc. These features will also be used in later stages. So far we get a candidate set $S_{reduced1}$ with a reasonable size, we still need to introduce more lexical and semantic features more suitable for STC task.

2.3 Ranking Stage I

2.3.1 Features

We first introduce seven more features which we believe can somehow capture the structures in matching texts in STC task, listed:

- cosine similarity between TF-IDF [6] vector of query and post
- cosine similarity between TF-IDF vector of query and comment
- cosine similarity between TF-IDF vector of query and post + comment
- negative Word Mover Distance [3] between query and post
- negative Word Mover Distance between query and comment
- negative Word Mover Distance between query and post + comment
- translation based language model [1] score $Score_{trans}$ between query and comment

2.3.2 Ranking

Then we perform ranking to further reduce $S_{reduced1}$. First, we treat each feature listed above as a simple ranker; the weight is either 1 or -1 based on experience. We regard the average rankings of each feature for each post-comment pair as the ranking score. Then, we rank the candidate post-comment pairs using the score and keep top 50 post-comment pairs as a reduced candidate set $S_{reduced2}$.

2.4 Ranking Stage II

2.4.1 Features

Most of the features mentioned in previous steps are either generated by linear models or not elaborately designed for STC task. Those features may not be able to capture richer structures in matching texts in STC task. We introduce more deep neural network models to calculate four more matching scores between the post and the comments.

The first model is a simple MLP (multi-layer perceptron). We use the element-wise sum of word vectors of a sentence as the sentence's representation, then pass the post's representation and the comment's representation through the neural network to calculate a matching score $Score_{embd}$.

The second model combines Bidirectional LSTM and CNN to get the representation of a sentence, similar to the model described in [13]. We pass the post's representation and the comment's representation through another MLP to calculate another matching score $Score_{BiLSTM+CNN}$.

Both models described above are trained with a ranking-based objective. We employ a large margin objective defined on preference pairs in ranking. Given triples (x, y^+, y^-), with x matched y^+ better than y^-, the loss function is defined as follows:

$$L = max(0, 1 + s(x, y^-) - s(x, y^+)) \tag{1}$$

where s is the matching model, x is a post, y^+ is a corresponding comment to the post, y^- is a random comment sampled from the training repository. After crawling from Weibo[1] and cleaning the corpus, we get about 12,000 thousand post-comment pairs with a similar distribution to the given repository. We merge them into the given repository, getting the extended training repository $Repo_{extn}$ used in this step.

We apply seq2seq models used in our generation-based method to calculate another two kinds of matching scores: $Score_{S2S-p2c}$ and $Score_{S2S-c2p}$. Details will be described in Section 3.3.1.

[1]http://www.weibo.com/

2.4.2 *Ranking*

Now we have several more matching scores newly generated at this stage, together with all the features' scores aforementioned, we will try to ensemble them to get a final ranking model. Besides the given labeled post-comment pairs, we crawl another 30 thousand post-comment pairs from Weibo and label them by crowdsourcing, getting an extended training set for learning to rank. We experiment on Ranking SVM [2] and LambdaMART [11] as the learning to rank algorithm, and it turns out that LambdaMART performs significantly better. We use the trained model to rank the candidates in $S_{reduced2}$, keeping top 10 as the final result for our retrieval-based method.

2.5 Experiments

We submitted three runs for the retrieval-based method, the only difference between these runs is the hyper-parameter of the final ranking model. We choose the hyper-parameter based on the model performance on validation data using different evaluation measures. ***SG01-C-R1*** corresponds to nG@1, ***SG01-C-R2*** corresponds to nERR@10, ***SG01-C-R3*** corresponds to P+. The evaluation results are listed in Table 1.

Table 1: Submission results

Runs	nG@1	P+	nERR@10
SG01-C-R1	0.5355	0.6084	0.6579
SG01-C-R2	0.5168	0.5944	0.6461
SG01-C-R3	0.5048	0.6200	0.6663
SG01-C-G1	**0.5867**	**0.6670**	**0.7095**
SG01-C-G2	0.5483	0.6335	0.6783
SG01-C-G3	0.5633	0.6567	0.6947
SG01-C-G4	0.4483	0.5545	0.6129
SG01-C-G5	0.3820	0.5068	0.5596

We analyze the features' effectiveness in ranking candidate post-comment pairs and find that $Score_{BiLSTM+CNN}$ and $Score_{trans}$ are a little more important than other features.

3. GENERATION-BASED METHOD

Our generation-based method has two parts: first, we feed a post to different well-trained generative models to generate various candidate comments; second, we assign scores to all the candidate comments and rank them according to their scores. Next, we describe our generation-based method in detail.

3.1 Data Pre-processing

As our models are all word-based, we first perform Chinese word segmentation on each post-comment pair and replace spaces in the original text with commas. We limit the number of consecutively repeated tokens (including punctuations) to at most three by truncating. For example, " 哈哈。。。。。 " (Haha.....) will be truncated to "哈哈。。。 " (Haha...).

3.2 Generative Models

We take advantage of the encoder-decoder framework [9] and apply dynamic memory to the attention mechanism [14] to improve the coherence of generated sentences. We also propose a VAE (Variational Auto-Encoder) variant to model one-to-many post-comment pairs by introducing stochasticity. Now we present our generative models.

S2SAttn uses the basic encoder-decoder framework described in [9] and the attention mechanism [14] to model the post-comment pairs. Given a post $X = (x_1, x_2, x_3 ... x_T)$ and one of its comment $Y = (y_1, y_2, y_3 ... y_{T'})$, where x_t and y_t are the t-*th* token in the post and the comment respectively, the model predicts the conditional probability:

$$p(y_1, ... y_{T'} | x_1, ... x_T) = \prod_{t=1}^{T'} p(y_t | v, y_1, ... y_{t-1}) \qquad (2)$$

where v is the hidden state of the encoder at the last step, which will be used to initialize the hidden state of the decoder.

S2SAttn-addmem introduces dynamic memory to the attention mechanism of S2SAttn model. At each time step during decoding, we tail the decoder's output to encoder's outputs to acquire an augmented memory for computing attention vectors. At decoder time step t we have:

$$Mem_t = \begin{cases} CONCAT(Mem_{t-1}, dec_t), & (t > 0) \\ enc_{1\sim T}, & (t = 0) \end{cases} \qquad (3)$$

where Mem_t is the memory matrix that will be referenced by attention mechanism at time step t during decoding phase; $enc_{1\sim T}$ is the total outputs of encoding phase; $CONCAT$ is a concatenation operation through time dimension. The dynamic memory method takes into account the information that has already been generated to improve the coherence of generated sentences.

VAEAttn introduces a random variable z into the seq2seq framework, providing a stronger modeling capability for those pairs in which a variety of comments are found to one post. This randomness also allows the model to generate different valid comments for identical inputs. We take advantage of VAE to encode X into a probability distribution. For our generation task, however, we do not reconstruct X, instead we decode from z to get candidate comments:

$$p(y_1, ...y_{T'}|x_1, ...x_T) = \prod_{t=1}^{T'} p(y_t|z, y_1, ...y_{t-1}) \tag{4}$$

where z is sampled from the distribution based on encoding state s and parameterised by θ:

$$z \sim P_\theta(z, s) \quad s = encoder(x_1...x_T) \tag{5}$$

For simplicity, we suppose that the latent variable z satisfies the isolated Gaussian distribution with parameter (μ, Σ). Since the backpropagation process does not work with random variables, we use the reparameterization method to rewrite the sampling process as:

$$z' \sim normal(0, 1) \quad z = z' * \sigma(s) + \mu(s) \tag{6}$$

where $\mu(s)$ and $\sigma(s)$ are functions (e.g., an MLP) to encode s into the distribution parameters. To avoid over-fitting of the latent distribution, we also add a Kullback-Leibler divergence between the standard normal distribution and the latent variable distribution to the original sequence loss, which acts as a regularization term.

We use ***Segment-beam-search*** to further increase the diversity of generated sequences. In the generation process, we want to avoid homogeneous prefixes as far as possible. At each decoding step, the beams are divided into M segments, and each segment contains n_m beams that are used to construct a search space with size $n_m \times vocabulary\_size$, we then select top n_m candidates from this sub-space as a new group of beams. In this way, we can keep more different prefixes, introducing more semantic diversity into our generated candidates.

3.3 Scoring and Ranking

3.3.1 Scoring Models

Our well-trained seq2seq models (S2SAttn and S2SAttn-addmem) are potentially scorers. Given a post, we have as many post-comment pairs as generated candidates on which can we use the scorers to compute two kinds of logarithmic probability described as follows:

$Score_{S2S-p2c}$ is a prediction of logarithmic P(Y|X) which is also known as *likelihood*. The given post-comment pair is sent to well-trained seq2seq models just as the same way done during the training process; the difference is that we compute the cross-entropy loss without any backpropagation. We then accumulate these losses over the whole sequence as our final score.

$Score_{S2S-c2p}$ is a prediction of logarithmic P(X|Y) which is also known as *posterior*. It is computed almost the same way as $Score_{S2S-p2c}$, the difference is that the seq2seq models it uses are trained by comment-post pairs.

$Score_{S2S-p2c}$ and $Score_{S2S-c2p}$ will be combined later for candidates selection.

3.3.2 Filter by Rules

We filter the generated candidates from the following three aspects:

- Maintain a blacklist, which is used to exclude candidates that are not proper for chatting due to some content bias of the source of the corpus, such as "关注我的微博" (Follow my Weibo). We also exclude candidates containing UNK (tokens not in vocabulary).
- Deduplicate consecutively repeated sentence segments. For example, "是啊，你说得太好了，你说得太好了！" (Yes, you are right, you are right!) will be truncated to "是啊，你说得太好了！" (Yes, you are right!).
- Truncate consecutively repeated punctuations in a sentence, leaving them only up to three consecutive appearances. For example, we truncate "!!!!!" to "!!!". This rule is only for punctuations other than else tokens.

3.3.3 Ranking

We apply likelihood and posterior measures described in Section 3.3.1 to rank the generated candidates. We combine likelihood and posterior scores and then discount it with a ratio based on comment's length.

First we collect scores derived from different seq2seq models, to get likelihood score Li and posterior score Po:

$$Li = \sum_k (score_{S2S-p2c})_k \tag{7}$$

$$Po = \sum_k (score_{S2S-c2p})_k \tag{8}$$

We use lp as a discount ratio described in [12], it is used to alleviate the bias caused by sequence length in machine translation task. For a generated candidate Y' of length $|Y'|$ with a degree parameter α, lp is computed as:

$$lp(Y') = \frac{(c + |Y'|)^\alpha}{(c + 1)^\alpha} \tag{9}$$

The overall ranking score for a given post-comment pair is defined as:

$$score = \frac{\lambda * Li + (1 - \lambda) * Po}{lp(Y')} \tag{10}$$

we use this score to rank generated candidates, getting top 10 comments for each query.

3.4 Experiments

3.4.1 Implementation and Submissions

We use the dynamic RNN implementation in Tensorflow[2] to build models described above. A three-layer LSTM cell with 1024 dimensions is adopted for both encoder and decoder. The word embedding is learned during training, which is initialized with 150 dimensions by a pre-trained one. We train the generative models on $Repo_{extn}$ aforementioned using the Adam optimizer with an initial learning rate 10e-4.

We made the following five runs by a combination of candidates from different models and scoring methods.

- ***SG01-C-G5*** is a combination of candidates from VAEAttn and VAEAttn-addmem, scored only by Li, discounted by lp.
- ***SG01-C-G4*** is a combination of candidates from S2SAttn and S2SAttn-addmem, scored only by Li, discounted by lp.
- ***SG01-C-G3*** is a combination of candidates from S2SAttn and S2SAttn-addmem, scored by Li and Po, discounted by lp.
- ***SG01-C-G2*** is a combination of candidates from VAEAttn and VAEAttn-addmem, scored by Li and Po, discounted by lp.
- ***SG01-C-G1*** is a combination of all candidates from all models, scored by Li and Po, discounted by lp.

All the candidates are generated using segmented-beam-search. The submission results are included in Table 1.

3.4.2 Case Study

We show some cases as a supplement to our experiments to reveal more about how those improvements we made to baseline models benefit candidates generation and ranking.

Table 2 shows some results from SG01-C-G3 and SG01-C-G4 as a comparison to give an intuitive understanding of how the feature Po works. We see that by incorporating Po into ranking criteria with a proper coefficient λ, we can suppress some frequent but less informative comments while doing little harm to those less frequent but coherent ones.

Table 3 shows some results from SG01-C-G1, SG01-C-G2, and SG01-C-G3. We observe that comments generated by VAE models are quite different from those by seq2seq models, by being somewhat more specific and covering more aspects of the topic, however, sometimes their quality may not be stable due to its randomness. As for SG01-C-G1 results, which is a fusion of results from different models, we find that some preferred candidates from SG01-C-G2 and SG01-C-G3 are gathered into SG01-C-G1, taking a higher rank in its candidate list. This might shed some light on why SG01-C-G1 achieves a better outcome than other runs.

[2]https://www.tensorflow.org/

Table 2: Case study 1

Query 和家人一起喝喝茶，聊聊天，也是一种生活的乐趣 (Drink tea and chat with the family, what a joy of life)
Related comment-list from SG01-C-G3
我也是这样觉得 (I feel the same)
我也在看呢 (I'm watching too)
是啊，生活是一种享受 (Yes, life is joyful)
我也是。。。 (Me too...)
是的，我也这么认为 (Yes, I also believe so)
我也是!!! (Me too!!!)
呵呵，是啊！ (Uh, yeah!)
是啊是啊！ (Yeah, yeah!)
是的，是的。 (Yes, yes.)
我也是这么想的 (I think so, too)
Related comment-list from SG01-C-G4
是的，是的。 (Yes, yes.)
我也是。。。 (Me too...)
我也是这么想的 (I think so, too)
我也是!!! (Me too!!!)
是啊，生活是一种享受 (Yes, life is joyful)
是啊是啊！ (Yeah, yeah!)
我也是这样觉得 (I feel the same)
是的，我也这么认为 (Yes, I also believe so)
呵呵，是啊！ (Uh, yeah!)
我也在看呢 (I'm watching too)

4. CONCLUSIONS

We perform statistical significance experiments on all eight submissions; the results are listed in Table 4 and Table 5. Along with Table 1, we can observe that generation-based method can achieve better performance under all the evaluation measures.

We think there are some reasons for this: although we make efforts to generate longer and more informative comments, it turns out that the generation-based method still gives shorter and less informative comments than the retrieve-based method. Although there is punishment for less informative comments, the generation-based method can

Table 3: Case study 2

Query 杭州的亲们，我们已登机，等待起飞啦，暂别数日。 (My dear friends in Hangzhou, we are on board, waiting for take off, won't be seeing you for a while.)
Related comment-list from SG01-C-G1
辛苦了,注意安全! (You've had a long day, be safe!)
辛苦了。。。 (You've had a long day...)
也祝您节日快乐! (Wish you a happy holiday, too!)
一定要注意安全啊! (Must be safe!)
去哪啊? (Where are you going?)
一路平安,注意安全啊。。。 (Have a good trip, be safe...)
你要去哪里啊? (Where are you going?)
一路平安!!! (Have a good trip!!!)
祝您旅途愉快! (Wish you a happy journey!)
我也在等飞机。。。 (I'm also waiting for boarding...)
Related comment-list from SG01-C-G2
也祝您节日快乐! (Wish you a happy holiday, too!)
一定要注意安全啊! (Must be safe!)
祝您旅途愉快! (Wish you a happy journey!)
杭州欢迎您! (Welcome to Hangzhou!)
杭州欢迎你! (Welcome to Hangzhou!)
回杭州了吗? (Back to Hangzhou?)
什么时候来杭州啊? (When coming to Hangzhou?)
来杭州了? (Coming to Hangzhou?)
这么晚还不睡啊 (It's been late, still up?)
必须来支持! 加油! (Will support you! Good luck!)
Related comment-list from SG01-C-G3
辛苦了,注意安全! (You've had a long day, be safe!)
去哪啊? (Where are you going?)
辛苦了。。。 (You've had a long day...)
你要去哪里啊? (Where are you going?)
一路平安,注意安全啊。。。 (Have a good trip, be safe...)
一路平安!!! (Have a good trip!!!)
我也在等飞机。。。 (I'm also waiting for boarding...)
好的，等你消息。 (Okay, wait for your message.)
谢谢亲们的支持! (Thank you for your support!)
好的，谢谢! (Okay, thanks!)

still benefit from the evaluation criteria. Meanwhile, the given repository for the retrieval-based method is quite small for an open domain STC task, which makes it more challenging to avoid selecting comments not logically coherent with the query.

This paper has presented how we solve the STC problem with the retrieval-based method and the generation-based method. In the retrieve-based method, we bring in DNN and seq2seq scores as well as traditional text features to discover the valuable information for candidates selection. In the generation-based method, we improve seq2seq to promote diversity of candidates and make the most use of kinds

Table 4: Statistical signicance with each run of S-G01 according to the Official STC-2 Chinese performances (Randomised Tukey HSD test, B = 10000, $\alpha = 0.05$)

Terms	These runs are	Significantly better than these runs
Mean nG@1	SG01-C-G1	SG01-C-G4, SG01-C-G5
	SG01-C-G3	SG01-C-G5
	SG01-C-G2	SG01-C-G5
	SG01-C-R1	SG01-C-G5
Mean P+	SG01-C-G1	SG01-C-G4, SG01-C-G5
	SG01-C-G3	SG01-C-G4, SG01-C-G5
	SG01-C-G2	SG01-C-G5
	SG01-C-R3	SG01-C-G5
Mean nERR@10	SG01-C-G1	SG01-C-G5
	SG01-C-G3	SG01-C-G5
	SG01-C-G2	SG01-C-G5
	SG01-C-R3	SG01-C-G5
	SG01-C-R1	SG01-C-G5

Table 5: Statistical significance with each run of S-G01 according to the Unanimity-Aware (p = 0.2) STC-2 Chinese performances (Randomised Tukey HSD test, B = 10000, $\alpha = 0.05$)

Terms	These runs are	Significantly better than these runs
Mean nG@1	SG01-C-G1	SG01-C-G5
	SG01-C-G3	SG01-C-G5
	SG01-C-G2	SG01-C-G5
	SG01-C-R1	SG01-C-G5
Mean P+	SG01-C-G1	SG01-C-G4, SG01-C-G5
	SG01-C-G3	SG01-C-G5
	SG01-C-G2	SG01-C-G5
	SG01-C-R3	SG01-C-G5
	SG01-C-R1	SG01-C-G5
Mean nERR@10	SG01-C-G1	SG01-C-G5
	SG01-C-G3	SG01-C-G5
	SG01-C-G2	SG01-C-G5
	SG01-C-R3	SG01-C-G5
	SG01-C-R1	SG01-C-G5

of well-trained models for ranking. Finally, we achieve top performance in both methods.

We plan to investigate how to combine the retrieval-based method and the generation-based method; we also plan to introduce recent sophisticated models to our system to get better performance.

5. ACKNOWLEDGEMENTS

Our thanks to NTCIR-13 task organizers for their hard work, especially thanks to Noah' s Ark Lab for their work on organizing NTCIR-13 STC-2 Chinese subtask.

6. REFERENCES

[1] Z. Ji, Z. Lu, and H. Li. An information retrieval approach to short text conversation. *CoRR*, abs/1408.6988, 2014.

[2] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 133–142, New York, NY, USA, 2002. ACM.

[3] M. J. Kusner, Y. Sun, N. I. Kolkin, and K. Q. Weinberger. From word embeddings to document distances. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 957–966. JMLR.org, 2015.

[4] D. Metzler and W. B. Croft. A markov random field model for term dependencies. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 472–479, New York, NY, USA, 2005. ACM.

[5] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, pages 232–241, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[6] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5):513 – 523, 1988.

[7] L. Shang, T. Sakai, H. Li, R. Higashinaka, Y. Miyao, Y. Arase, and M. Nomoto. Overview of the NTCIR-13 Short Text Conversation Task. In *Proceedings of NTCIR-13*, 2017.

[8] L. Shang, T. Sakai, Z. Lu, H. Li, R. Higashinaka, and Y. Miyao. Overview of the ntcir-12 short text conversation task. In *NTCIR*, 2016.

[9] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc., 2014.

[10] T. Tao and C. Zhai. An exploration of proximity measures in information retrieval. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 295–302, New York, NY, USA, 2007. ACM.

[11] Q. Wu, C. J. C. Burges, K. M. Svore, and J. Gao. Adapting boosting for information retrieval measures. *Information Retrieval*, 13(3):254–270, Jun 2010.

[12] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, L. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016.

[13] R. Yan, Y. Song, X. Zhou, and H. Wu. "Shall I Be Your Chat Companion?": Towards an Online Human-Computer Conversation System. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, pages 649–658, New York, NY, USA, 2016. ACM.

[14] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo. Image captioning with semantic attention. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.