Dialogue Session Segmentation by Embedding-Enhanced TextTiling

Yiping Song,[1] Lili Mou,[1,2] Rui Yan,[1,3,4,] Li Yi,[4] Zinan Zhu,[4] Xiaohua Hu,[4] Ming Zhang[1]*

[1]Peking University, Beijing, China
[2]Key Laboratory of High Confidence Software Technologies (Peking University), MoE, China
[3]Natural Language Processing Department, Baidu Inc., Beijing, China *Corresponding author
[4]School of Computer, Central China Normal University, Wuhan, Hubei, China

songyiping@pku.edu.cn, doublepower.mou@gmail.com
{yanrui,yili,zzn,huxiaohua}@mail.ccnu.edu.cn

Abstract

In human-computer conversation systems, the context of a user-issued utterance is particularly important because it provides useful background information of the conversation. However, it is unwise to track all previous utterances in the current session as not all of them are equally important. In this paper, we address the problem of session segmentation. We propose an embedding-enhanced TextTiling approach, inspired by the observation that conversation utterances are highly noisy, and that word embeddings provide a robust way of capturing semantics. Experimental results show that our approach achieves better performance than the TextTiling, MMD approaches.

Index Terms: session segmentation, conversation system, word embeddings

1. Introduction

Human-computer dialog/conversation[1] is one of the most challenging problems in artificial intelligence. Given a user-issued utterance (called a *query* in this paper), the computer needs to provide a reply to the query. In early years, researchers have developed various domain-oriented dialogue systems, which are typically based on rules or templates [5, 6, 7]. Recently, open-domain conversation systems have attracted more and more attention in both academia and industry (e.g., XiaoBing from Microsoft and DuMi from Baidu). Due to high diversity, we can hardly design rules or templates in the open domain. Researchers have proposed information retrieval methods [8] and modern generative neural networks [9, 10] to either search for a reply from a large conversation corpus or generate a new sentence as the reply.

In open-domain conversations, context information (one or a few previous utterances) is particularly important to language understanding [2, 10, 11, 12]. As dialogue sentences are usually casual and short, a single utterance (e.g., "Thank you." in Figure 1) does not convey much meaning, but its previous utterance ("...writing an essay") provides useful background information of the conversation. Using such context will certainly benefit the conversation system.

However, tracking all previous utterances as the context is unwise. First, commercial chat-bots usually place high de-
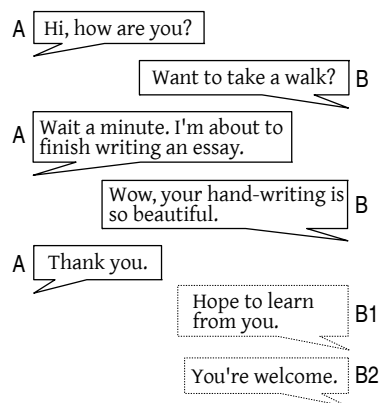

Figure 1: An example of multiple-turn dialogues.

mands on efficiency. In a retrieval-based system, for example, performing a standard process of candidate retrieval and re-ranking for each previous utterance may well exceed the time limit (which is very short, e.g., 500ms). Second, we observe that not all sentences in the current conversation session are equally important. The sentence "Want to take a walk?" is irrelevant to the current context, and should not be considered when the computer synthesizes the reply. Therefore, it raises the question of *session segmentation* in conversation systems.

Document segmentation for general-purpose corpora has been widely studied in NLP. For example, Hearst [13] proposes the TextTiling approach; she measures the similarity of neighboring sentences based on bag-of-words features, and performs segmentation by thresholding. However, such approaches are not tailored to the dialogue genre and may not be suitable for conversation session segmentation.

In this paper, we address the problem of session segmentation for open-domain conversations. We leverage the classic TextTiling approach, but enhance it with modern embedding-based similarity measures. Compared with traditional bag-of-words features, embeddings map discrete words to real-valued vectors, capturing underlying meanings in a continuous vector space; hence, it is more robust for noisy conversation corpora. Further, we propose a tailored method for word embedding learning. In traditional word embedding learning, the interaction between two words in a query and a reply is weaker than that within an utterance. We propose to combine a query and its corresponding reply as a "virtual sentence," so that it provides a better way of modeling utterances between two agents.

---

[1]A full dialog system typically involves speech recognition, text understanding, and speech synthesis. In this paper, we focus on the text understanding stage. However, our approach is directly applicable to dialogue systems with acoustic interaction, provided that the spoken language is converted to texts by automatic speech recognition (ASR) [1, 2] , or even manually text-transcribed for research purposes like [3, 4].

2. Related Work

2.1. Dialogue Systems and Context Modeling

Human-computer dialogue systems can be roughly divided into several categories. Template- and rule-based systems are mainly designed for certain domains [5, 6, 14]. Although manually engineered templates can also be applied in the open domain like [15], but their generated sentences are subject to 7 predefined forms, and hence are highly restricted. Retrieval methods search for a candidate reply from a large conversation corpus given a user-issued utterance as a query [8]. Generative methods can synthesize new replies by statistical machine translation [16, 17] or neural networks [9].

The above studies do not consider context information in reply retrieval or generation. However, recent research shows that previous utterances in a conversation session are important because they capture rich background information. Sordoni et al. [12] summarize a single previous sentence as bag-of-words features, which are fed to a recurrent neural network for reply generation. Serban et al. [18] design an attention-based neural network over all previous conversation turns/rounds, but this could be inefficient if a session lasts long in real commercial applications. By contrast, our paper addresses the problem of session segmentation so as to retain near, relevant context utterances and to eliminate far, irrelevant ones.

A similar (but different) research problem is topic tracking in conversations, e.g., [19, 20, 21, 22]. In these approaches, the goal is typically a classification problem with a few predefined conversation states/topics, and hence it can hardly be generalized to general-purpose session segmentation.

2.2. Text Segmentation

An early and classic work on text segmentation is TextTiling, proposed in [13]. The idea is to measure the similarity between two successive sentences with smoothing techniques; then segmentation is accomplished by thresholding of the depth of a "valley." In the original form of TextTiling, the cosine of term frequency features is used as the similarity measure. Joty et al. [23] apply divisive clustering instead of thresholding for segmentation. Malioutov et al. [24] formalize segmentation as a graph-partitioning problem and propose a minimum cut model based on $tf \cdot idf$ features to segment lectures. Ye et al. [25] minimize between-segment similarity while maximizing within-segment similarity. However, the above complicated approaches are known as *global* methods: when we perform segmentation between two successive sentences, future context information is needed. Therefore, they are inapplicable to real-time chat-bots, where conversation utterances can be viewed as streaming data.

In our study, we prefer the simple yet effective TextTiling approach for open-domain dialogue session segmentation, but enhance it with modern advances of word embeddings, which are robust in capturing semantics of words. We propose a tailored algorithm for word embedding learning by combining a query and context as a "virtual document"; we also propose several heuristics for similarity measuring.

3. Session Segmentation Methodology

3.1. TextTiling

We apply a TextTiling-like algorithm for session segmentation. The original TextTiling is proposed by Hearst [13]. The main idea is to measure the similarity of each adjacent sentence pair;
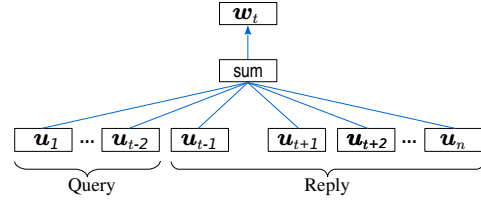

Figure 2: Word embedding learning by the continuous bag-of-words model with virtual sentences (the concatenation of a query and its reply). w_t is a word in the virtual sentence, either appearing in the query or the reply; the summed embeddings of remaining words are context.

then "valleys" of similarities are detected for segmentation.

Concretely, the "depth of the valley" is defined by the similarity differences between the peak point in each side and the current position. We may obtain some statistics of depth scores like the mean μ and standard deviation σ, and perform segmentation by a cutoff threshold.

$$cutoff(\alpha) = \mu + \alpha \cdot \sigma \qquad (1)$$

where α is a hyperparameter adjusting the number of segmentation boundaries; μ and σ are the average and standard deviation of depth scores, respectively.

In the scenario of human-computer conversations, we compute the depth solely by the similarity difference between its left peak (previous context) and the current position. This is because we cannot obtain future utterances during online conversation.

Although bag-of-words features work well in the original TextTiling algorithm for general text segmentation, it is not suitable for dialogue segmentation. As argued by Hearst [13], text overlap (repetition) between neighboring sentences is a strong hint of semantic coherence, which can be well captured by term frequency or $tf \cdot idf$ variants. However, in human-computer conversations, sentences are usually short, noisy, highly diversified, and probably incomplete, which requires a more robust way of similarity measuring. Therefore, we enhance TextTiling with modern word embedding techniques, as will be discussed in the next part.

3.2. Learning Word Embeddings

Word embeddings are distributed, real-valued vector representations of discrete words [26, 27]. Compared with one-hot representation, word embeddings are low-dimensional and dense, measuring word meanings in a continuous vector space. Studies show that the offset of two words' embeddings represents a certain relation, e.g., `man` − `woman` ≈ `king` − `queen` [26]. Hence, it is suitable to use word embeddings to model short and noisy conversation utterances.

To train the embeddings, we adopt the `word2vec` approach. The idea is to map a word w and its context c to vectors ($\boldsymbol{w}$ and $\boldsymbol{c}$). Then we estimate the probability of a word by

$$p(w|c) = \frac{\exp(\boldsymbol{w}^\top \boldsymbol{c})}{\sum_{\boldsymbol{w}'} \exp(\boldsymbol{w}'^\top \boldsymbol{c})} \qquad (2)$$

The goal of word embedding learning is to maximize the average probability of all words (suppose we have T running words):

$$\frac{1}{T} \sum_{t=1}^{T} \log p(w_t|c_t) \qquad (3)$$

We used hierarchical softmax to approximate the probability.

To model the context, we further adopt the continuous bag-of-words (CBOW) method. The context[2] is defined by the sum of neighboring words' (input) vectors in a fixed-size window ($t - \tau$ to $t + \tau$) within a sentence:

$$c_t = \sum_{\substack{t - \tau \leq i \leq t + \tau \\ i \neq t}} u_i \qquad (4)$$

Notice that the context vector u in Equation (4) and the output vector w in Equation (2) are different as suggested in [26, 27], but the details are beyond the scope of our paper.

Virtual Sentences

In a conversation corpus, successive sentences have a stronger interaction than general texts. For example, in Figure 1, the words *thank* and *welcome* are strongly correlated, but they hardly appear in the a sentence and thus a same window. Therefore, traditional within-sentence CBOW may not capture the interaction between a query and its corresponding reply.

In this paper, we propose the concept of virtual sentences to learn word embeddings for conversation data. We concatenate a query q and its reply r as a *virtual sentence* $q \oplus r$. We also use all words (other than the current one) in the virtual sentence as context (Figure 2). Formally, the context c_t of the word w_t is given by

$$c_t = \sum_{\substack{i \in q \oplus r \\ i \neq t}} u_i \qquad (5)$$

In this way, related words across two successive utterances from different agents can have interaction during word embedding learning. As will be shown in Subsection 4.2, virtual sentences yield a higher performance for dialogue segmentation.

3.3. Measuring Similarity

In this part, we introduce several heuristics of similarity measuring based on word embeddings. Notice that, we do not leverage supervised learning (e.g., full neural networks for sentence paring [28, 29]) to measure similarity, because it is costly to obtain labeled data of high quality.

The simplest approach, perhaps, is to sum over all word embeddings in an utterance as sentence-level features s. This heuristic is essentially the sum pooling method widely used in neural networks [30, 31, 28]. The cosine measure is used as the similarity score between two utterances S_1 and S_2. Let s_1 and s_2 be their sentence vectors; then we have

$$\text{sim}(S_1, S_2) = \cos(s_1, s_2) \equiv \frac{s_1^\top s_2}{\|s_1\| \cdot \|s_2\|} \qquad (6)$$

where $\| \cdot \|$ is the ℓ_2-norm of a vector.

To enhance the interaction between two successive sentences, we propose a more complicated heuristic as follows. Let w_i and v_j be a word in s_1 and s_2, respectively. (Embeddings are denoted as bold alphabets.) Suppose further that n_1 and n_2 are the numbers of words in S_1 and S_2. The similarity is given by

$$\text{sim}(S_1, S_2) = \frac{1}{n_1} \sum_{i=1}^{n_1} \max_{j=0}^{n_2} \{\cos(w_i, v_j)\} \qquad (7)$$

For each word w_i in s_1, our intuition is to find the most related word in s_2, given by the $\max\{\cdot\}$ part; their relatedness

Method	P	R	F
Random	36.9	32.0	34.2
MMD	27.0	23.0	25.0
TextTiling with *tf·idf*	66.5	44.8	53.5
+ embeddings trained by virtual sentences, and *heuristic-max* similarity	**71.7**	**78.0**	**74.7**

Table 1: Dialogue session segmentation performance in terms of precision (P), recall (R) and F-measure (F). Results are in percentage.

is also defined by the cosine measure. Then the sentence-level similarity is obtained by the average similarity score of words in s_1. This method is denoted as ***heuristic-max***.

Alternatively, we may substitute the max operator in Equation (7) with avg, resulting in the ***heuristic-avg*** variant, which is equivalent to the average of word-by-word cosine similarity. However, as shown in Subsection 4.2, intensive similarity averaging has a "blurring" effect and will lead to significant performance degradation. This also shows that our proposed *heuristic-max* does capture useful interaction between two successive utterances in a dialogue.

4. Experiments

In this section, we evaluate our embedding-enhanced TextTiling method as well as the effect of session segmentation. In Subsection 4.1, we describe the datasets used in our experiments. Subsection 4.2 presents the segmentation accuracy of our method and baselines. In Subsection 4.3, we show that, with our session segmentation, we can improve the performance of a retrieval-based conversation system.

4.1. Dataset

To evaluate the session segmentation method, we used a real-world chatting corpus from DuMi,[3] a state-of-the-practice open-domain conversation system in Chinese. We sampled 200 sessions as our experimental corpus. Session segmentation was manually annotated before experiments, serving as the ground truth. The 200 sessions were randomly split by 1:1 for validation and testing. Notice that, our method does not require labeled training samples; massive data with labels of high quality are quite expensive to obtain.

We also leveraged an unlabeled massive dataset of conversation utterances to train our word embeddings with "virtual sentences." The dataset was crawled from the Douban forum,[4] containing 3 million utterances and approximately 150,000 unique words (Chinese terms).

4.2. Segmentation Performance

We compared our full method (TextTiling with *heuristic-max* based on embeddings trained by virtual sentences) with several baselines:

- *Random.* We randomly segmented conversation sessions. In this baseline, we were equipped with the prior probability of segmentation.
- *MMD.* We applied the MinMax-Dotplotting (MMD) approach proposed by Ye et al. [25]. We ran the executable program provided by the authors.

---

[2]Here, the context of a word roughly refers to its previous and future words. Please do not be confused with the context of an utterance.

[3]http://xiaodu.baidu.com
[4]http://www.douban.com

Embeddings	sum pooling			*heuristic-max*			*heuristic-avg*		
	P	R	F	P	R	F	P	R	F
Virtual-sentence	71.6	78.4	**74.8**	71.7	78.0	<u>74.7</u>	64.5	67.8	<u>66.1</u>
Within-sentence	66.1	73.1	69.4	70.0	74.3	**72.1**	62.7	67.4	65.0
Window context	65.6	69.5	67.5	70.7	76.8	**73.6**	62.2	65.4	63.8

Table 2: Analysis of word embedding strategies and similarity heuristics. Bold numbers are the highest value in each row; underlined ones are the highest in each column.

Method	p@1	nDCG
Fixed context	0.484	0.699
Session segmentation	0.521	0.737

Table 3: A retrieval dialogue system with fixed context (2 previous utterances) and the proposed sentence segmentation (virtual sentences with *heuristic-max*).

- *TextTiling w/ tf·idf features.* We implemented TextTiling ourselves according to [13].

We tuned the hyperparameter α in Equation (**??**)on the validation set to make the number of segmentation close to that of manual annotation, and reported precision, recall, and the F-score on the test set in Table 1. As seen, our approach significantly outperforms baselines by a large margin in terms of both precision and recall. Besides, we can see that MMD obtains low performance, which is mainly because the approach cannot be easily adapted to other datasets like short sentences of conversation utterances. In summary, we achieve an F-score higher than baseline methods by more than 20%, showing the effectiveness of enhancing TextTiling with modern word embeddings.

We further conducted in-depth analysis of different strategies of training word-embeddings and matching heuristics in Table 2. For word embeddings, we trained them on the 3M-sentence dataset with three strategies: (1) virtual-sentence context proposed in our paper; (2) within-sentence context, where all words (except the current one) within a sentence (either a query or reply) are regarded as the context; (3) window-based context, which is the original form of [26]: the context is the words in a window (previous 2 words and future 2 words in the sentence). We observe that our virtual-sentence strategy consistently outperforms the other two in all three matching heuristics. The results suggest that combining a query and a reply does provide more information in learning dialogue-specific word embeddings.

Regarding matching heuristics, we find that in the second and third strategies of training word embeddings, the complicated *heuristic-max* method yields higher F-scores than simple sum pooling by 2–3%. However, for the virtual-sentence strategy, *heuristic-max* is slightly worse than the sum pooling. (The degradation is only 0.1% and not significant.) This is probably because both *heuristic-max* and virtual sentences emphasize the rich interaction between a query and its corresponding reply; combining them does not result in further gain.

We also notice that *heuristic-avg* is worse than other similarity measures. As this method is mathematically equivalent to the average of word-by-word similarity, it may have an undesirable blurring effect.

To sum up, our experiments show that both the proposed embedding learning approach and the similarity heuristic are effective for session segmentation. The embedding-enhanced TextTiling approach largely outperforms baselines.

4.3. Session Segmentation in Dialogue Systems

We conducted an external experiment to show the effect of session segmentation in dialogue systems. We integrated the segmentation mechanism into a state-of-the-practice retrieval-based system and evaluated the results by manual annotation, similar to our previous work [28, 32, 33].

Concretely, we compared our session segmentation with fixed-length context, used in [12]. That is to say, the competing method always regards two previous utterances as context. We hired three workers to annotate the results with three integer scores (0–2 points, indicating bad, borderline, and good replies, respectively.) We sampled 30 queries from the test set of 100 sessions. For each query, we retrieved 10 candidates and computed p@1[5] and nDCG scores [34] (averaged over three annotators). Provided with previous utterances as context, each worker had up to 1000 sentences to read during annotation.

Table 3 presents the results of the dialogue system with session segmentation. As demonstrated, our method outperforms the simple fixed-context approach in terms of both metrics. We computed the inner-annotator agreement: std = 0.309; 3-discrete-class Fleiss' kappa score = 0.411, indicating moderate agreement [35].

Case Study. We present a case study on our website: https://sites.google.com/site/sessionsegmentation/. From the case study, we see that the proposed approach is able to segment the dialogue session appropriately, so as to better utilize background information from a conversation session.

5. Conclusion

In this paper, we addressed the problem of session segmentation for open-domain dialogue systems. We proposed an embedding-enhanced TextTiling approach, where we trained embeddings with the novel notion of virtual sentences; we also proposed several heuristics for similarity measure. Experimental results show that both our embedding learning and similarity measuring are effective in session segmentation, and that with our approach, we can improve the performance of a retrieval-based dialogue system.

6. Acknowledgments

---

[5] Out of a rigorous criterion, we regard a "correct" reply if the score is 2, and "incorrect" if the score is 0 or 1.

7. References

[1] F. Bechet, A. Nasr, and B. Favre, "Adapting dependency parsing to spontaneous speech for open domain spoken language understanding." in *INTERSPEECH*, 2014, pp. 135–139.

[2] C. Liu, P. Xu, and R. Sarikaya, "Deep contextual language understanding in spoken dialogue systems," in *INTERSPEECH*, 2015, pp. 120–124.

[3] A. Cervone, C. Lai, S. Pareti, and P. Bell, "Towards automatic detection of reported speech in dialogue using prosodic cues," in *INTERSPEECH*, 2015, pp. 3061–3065.

[4] V. Freeman, G.-A. Levow, R. Wright, and M. Ostendorf, "Investigating the role of 'yeah' in stance-dense conversation," in *INTERSPEECH*, 2015, pp. 3076–3080.

[5] G. Ferguson, J. Allen, B. Miller *et al.*, "TRAINS-95: Towards a mixed-initiative planning assistant." in *AIPS*, 1996, pp. 70–77.

[6] A. C. Graesser, P. Chipman, B. C. Haynes, and A. Olney, "AutoTutor: An intelligent tutoring system with mixed-initiative dialogue," *IEEE Trans. Education*, vol. 48, no. 4, pp. 612–618, 2005.

[7] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding." in *INTERSPEECH*, 2013, pp. 3771–3775.

[8] J. Otterbacher, G. Erkan, and D. R. Radev, "Using random walks for question-focused sentence retrieval," in *HLT-EMNLP*, 2005, pp. 915–922.

[9] L. Shang, Z. Lu, and H. Li, "Neural responding machine for short-text conversation," in *ACL-IJCNLP*, 2015, pp. 1577–1586.

[10] A. Sordoni, Y. Bengio, H. Vahabi, C. Lioma, J. Grue Simonsen, and J.-Y. Nie, "A hierarchical recurrent encoder-decoder for generative context-aware query suggestion," in *CIKM*, 2015, pp. 553–562.

[11] A. Bhargava, A. Celikyilmaz, D. Hakkani-Tur, and R. Sarikaya, "Easy contextual intent prediction and slot detection," in *ICASSP*, 2013, pp. 8337–8341.

[12] A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J.-Y. Nie, J. Gao, and B. Dolan, "A neural network approach to context-sensitive generation of conversational responses," in *NAACL-HLT*, 2015, pp. 196–205.

[13] M. A. Hearst, "TextTiling: Segmenting text into multi-paragraph subtopic passages," *Computational Linguistics*, vol. 23, no. 1, pp. 33–64, 1997.

[14] S. Watanabe, J. R. Hershey, T. K. Marks, Y. Fujii, and Y. Koji, "Cost-level integration of statistical and rule-based dialog managers," in *INTERSPEECH*, 2014, pp. 323–327.

[15] S. Han, J. Bang, S. Ryu, and G. G. Lee, "Exploiting knowledge base to generate responses for natural language dialog listening agents," in *SIGDIAL*, 2015, pp. 129–133.

[16] C. Haas and S. Riezler, "Response-based learning for machine translation of open-domain database queries," in *NAACL-HLT*, 2015, pp. 1339–1344.

[17] J. Tavernier, R. Cowan, and M. Vanni, "Holy Moses! Leveraging existing tools and resources for entity translation," in *LREC*, 2008, pp. 2715–2719.

[18] I. V. Serban, A. Sordoni, Y. Bengio, A. Courville, and J. Pineau, "Building end-to-end dialogue systems using generative hierarchical neural network models," *arXiv preprint arXiv:1507.04808*, 2015.

[19] S. Kim, R. E. Banchs, and H. Li, "Wikipedia-based kernels for dialogue topic tracking," in *ICASSP*, 2014, pp. 131–135.

[20] A. Celikyilmaz, D. Z. Hakkani-Tür, and G. Tür, "Approximate inference for domain detection in spoken language understanding," in *INTERSPEECH*, 2011, pp. 713–716.

[21] K. Lagus and J. Kuusisto, "Topic identification in natural language dialogues using neural networks," in *SIGDIAL*, 2002, pp. 95–102.

[22] P. A. Crook, J.-P. Robichaud, and R. Sarikaya, "Multi-language hypotheses ranking and domain tracking for open domain dialogue systems," in *INTERSPEECH*, 2015, pp. 1810–1814.

[23] S. Joty, G. Carenini, and R. T. Ng, "Topic segmentation and labeling in asynchronous conversations," *JAIR*, no. 47, pp. 521–573, 2013.

[24] I. Malioutov and R. Barzilay, "Minimum cut model for spoken lecture segmentation," in *COLING-ACL*, 2006, pp. 25–32.

[25] N. Ye, N. Ye, J. Zhu, H. Wang, M. Y. Ma, and B. Zhang, "An improved model of dotplotting for text segmentation," *J. Chinese Language and Computing*, vol. 17, no. 1, pp. 27–40, 2007.

[26] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

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

[28] R. Yan, Y. Song, and H. Wu, "Learning to respond with deep neural networks for retrieval based human-computer conversation system," in *SIGIR*, 2016.

[29] L. Mou, R. Men, G. Li, Y. Xu, L. Zhang, R. Yan, and Z. Jin, "Natural language inference by tree-based convolution and heuristic matching," in *ACL (2)*, 2016.

[30] Y. LeCun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J. Denker, H. Drucker, I. Guyon, U. Muller, E. Sackinger, P. Simard, and V. Vapnik, "Comparison of learning algorithms for handwritten digit recognition," in *Proc. Int'l Conf. Artificial Neural Networks*, 1995, pp. 53–60.

[31] L. Mou, H. Peng, G. Li, Y. Xu, L. Zhang, and Z. Jin, "Discriminative neural sentence modeling by tree-based convolution," in *EMNLP*, 2015, pp. 2315–2325.

[32] X. Li, L. Mou, R. Yan, and M. Zhang, "StalemateBreaker: A proactive content-introducing approach to automatic human-computer conversation," in *IJCAI*, 2016.

[33] L. Mou, R. Yan, G. Li, L. Zhang, and Z. Jin, "Backward and forward language modeling for constrained natural language generation," *arXiv preprint arXiv:1512.06612*, 2015.

[34] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," *ACM Trans. Information Systems*, vol. 20, no. 4, pp. 422–446, 2002.

[35] J. Fleiss, "Measuring nominal scale agreement among many raters," *Psychological Bulletin*, vol. 76, no. 5, pp. 378–382, 1971.