IMPROVING THE ROBUSTNESS OF EXAMPLE-BASED DIALOG RETRIEVAL USING RECURSIVE NEURAL NETWORK PARAPHRASE IDENTIFICATION

Lasguido Nio, Sakriani Sakti, Graham Neubig, Tomoki Toda, Satoshi Nakamura

Graduate School of Information Science, Nara Institute of Science and Technology
8916-5 Takayama-cho, Ikoma, Nara 630-0192, Japan
E-mail: {lasguido.kp9,ssakti,neubig,tomoki,s-nakamura}@is.naist.jp

ABSTRACT

Previous work on example-based chat-oriented dialog systems utilizing real human-to-human conversation has shown promising results. However, most previous methods use relatively simple retrieval techniques, resulting in weakness to out of vocabulary (OOV) database queries and inadequate handling of interactions between words in the sentence. To overcome this problem, in this paper we propose a method to utilize recursive neural network paraphrase identification to improve the accuracy and robustness of example-based dialog response retrieval. We model our dialog-pair database and user input query with distributed word representations, and employ recursive autoencoders and dynamic pooling to determine whether two sentences with arbitrary length have the same meaning. The distributed representations have the potential to improve handling of OOV cases, and the recursive structure can reduce confusion in example matching. We evaluate the system performance based on objective and subjective metrics.

***Index Terms*—** example based dialog system, recursive neural network, paraphrase identification

1. INTRODUCTION

Data-driven approaches are seeing increasing interest as lightweight methods to create broad-coverage chat-oriented dialog systems [1, 2, 3, 4]. These approaches are attractive because, in contrast to rule-based approaches [5, 6], they allow for the use of large amounts of data on the Web to efficiently find responses for a large variety of user queries. In particular, the data used in these systems usually consists of query-response pairs, where the query is representative of the user's input to the system, and the response is representative of the system's response. To achieve broad coverage, recording of a large data set of real human-to-human conversation is necessary, and some studies propose constructing dialog examples from available log databases created using Wizard of OZ (WOZ) systems [7] or Twitter [8].

There are many approaches to data-driven dialog, and example based dialog modeling (EBDM) is one of them. By retrieving examples from a database and displaying the response to the user, EBDM is only able to generate examples that are actually included in the database. Because of this, it is able to generate highly natural output when a response is included in the database and the example is able to be appropriately retrieved [1, 2, 3], but if the system is not able to find similar examples to determine the response, most EBDM systems currently rely on canned or template responses which may result in less than satisfactory output [9, 10].

Response retrieval for EBDM works by matching the user's utterance with a query in the query-response database, then returning the response that corresponds with the most closely matching query. In the majority of previous work, this matching is performed by simple lexical measures such as TF-IDF weighted cosine similarity [11, 12] or syntactic-semantic similarity based on POS strings and WordNet synsets [13]. However, we can think of a number of situations in which these simplistic methods are clearly inadequate. For example, if the user's utterance is "it is not raining today," previously proposed matching methods will give "it is raining today" a high score, and the system may provide the exact opposite response a user desires. In general, two factors contribute greatly to the accuracy of EBDM systems: the coverage of the dialogue corpus, and the effectiveness of the example retrieval. In this paper, we focus particularly on the latter of these problems in EBDM systems, arguing that more sophisticated methods for matching user utterances and queries in the database are necessary.

In this work, we focus on the recent great improvements in distributional representations of language using vector space word representations [14]. In particular, we note that compositional distributional representations using neural networks [15] have the potential to capture a large number of linguistic phenomena, such as the above-mentioned inversion, or simpler phenomena such as paraphrases. We learn these representations and use them to retrieve an appropriate response given a user utterance based on the paraphrase detection model of Socher et al. [16]. As a testbed for the proposed method, we utilize dialog corpora constructed from movie conversation data, and examine its effectiveness compared to a previously proposed method.

2. BASELINE EBDM

In general, EBDM chooses a response from the examples stored in the dialog database. In order to do so, it computes a similarity measure between the user input and the query part of the query-response pairs, and returns the associated response for the query with the highest similarity. We utilize TF-IDF based cosine similarity [17, 18] as our baseline similarity measure for use in EBDM.

User input is treated by the dialog management system as a query to the response generator module. Given the user input, our response generator module will search for an appropriate response through the entire example database. The response generator first searches through the database with TF-IDF based cosine similarity retrieval. If no example with sufficiently high similarity is found, the response generator has encountered an OOV problem and will output a potentially uncorrelated utterance, or fall back to a canned response.

TF-IDF based cosine similarity calculates cosine similarity (Equation (1)) over the term vector of two sentences, S_1 and S_2. We construct the term vector by applying additional TF-IDF weighting (Equation (2)) to increase the emphasis on important words [19]. We set $F_{t,T}$ as a term frequency t in a sentence T, and DF_t as the total number of sentences in the query-response pairs that contain term t.

$$cos_{sim}(S_1, S_2) = \frac{S_1 \cdot S_2}{\parallel S_1 \parallel \parallel S_2 \parallel} \tag{1}$$

$$TFIDF(t, T) = F_{t,T} \log \left(\frac{|T|}{DF_t} \right) \tag{2}$$

3. NEURAL NETWORK BASED RETRIEVAL

However, simple methods such as cosine similarity have problems with robustness, which we will discuss in more detail in the following sections. In this section, we describe our proposed method to use neural network-based retrieval to retrieve more appropriate responses from the example database. In this method, a proper system response is retrieved by modeling our example database using neural word representations and calculating the probability that the user input and a query in the database are paraphrases.

Adopting the work of [16], we utilize recursive autoencoders (RAE), dynamic pooling, and a softmax classifier to decide weather the sentence is paraphrased or not. In the following sections we describe about: (1) word representations, the input to the RAE, (2) recursive autoencoders, and (3) dynamic pooling and paraphrase classification. An overview of the neural-network-based retrieval is depicted in Figure 1.

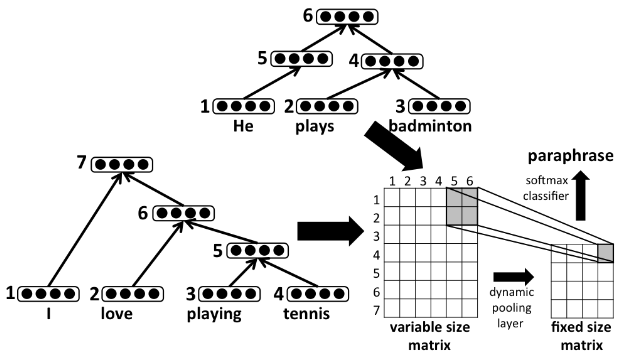


Fig. 1. Overview of neural-network-based retrieval.

3.1. Word Representations

A distributed word representation is an n-dimensional vector of continuous values used to represent a word in the vocabulary. They are often obtained by joint learning of neural network language models and distributed representation for words [20]. Improved word representations [14] are known to capture distributional syntactic and semantic information via the word co-occurrence statistics. In a word representation, each word in dictionary ($i \in D$) is embedded into n-dimensional space $L \in \mathbb{R}^{n \times |D|}$. From this representation, a word vector can be seen as a single vector in the column L.

The reason why word representations are useful is that they allow for soft matching of similar words when exact matches are no available. As an example, we can imagine there is an input sentence "I like to frequent taverns" which is unavailable in the example database. Given this sentence, the existing response generator will fail and response the user input with another uncorrelated response. However, if we use distributed word representations, we will also be able to match examples that use semantically similar but different words such as "visit" for "frequent" and "bars" for "taverns".

3.2. Recursive Autoencoder

The RAE algorithm is used to combine word representations into vector representations of longer phrases in a syntactic parse tree. The aim of using the syntactic parse tree is to capture the compositionality of meaning that is naturally constrained by the tree. In order to construct the vector representation, this algorithm requires word representations and a binary syntactic tree as input.

When calculating the recursive autoencoders, every child and non-terminal node in the binary tree is collected as a feature representation of a sentence. The binary tree forms the parent and children triplets ($p \rightarrow c_1 c_2$) where each child could be a word representation vector or other non-terminal nodes. A parent p is calculated through the neural network layer (Equation (3))

$$p = f(W_e[c_1; c_2] + b), \tag{3}$$

where $[c_1; c_2]$ is concatenation of the vector of two children and f is a tanh activation function.

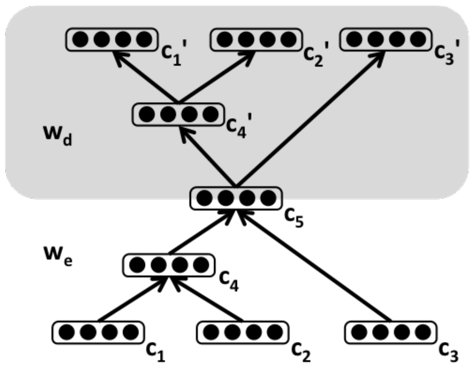


Fig. 2. Recursive autoencoder model.

The parameters W_e and b are trained using recursive autoencoders as shown in Figure 2. The RAE performance is evaluated through the Euclidean distance between original input and its estimated reconstruction node (Equation (4))

$$E(p) = ||[c_1; c_2] - [c'_1; c'_2]||^2 \tag{4}$$

where

$$[c'_1, c'_2] = f(W_d p + b_d). \tag{5}$$

This process will be repeated recursively for all non-terminal nodes. In the course of RAE training, we want to minimize the total error of all inputs pairs on every non-terminal node. The total error can be obtain by adding up all the calculated error from a single parse tree T

$$E_{tree}(T) = \sum_{p \in T} E(p). \tag{6}$$

The benefit of recursive autoencoder is that they can capture the compositional structure of phrases, and their similarity the two given sentences. For example, a sentence "tons of stuff to throw away" and "a lot of junk to dispose" there are relationship between words and phrases such as "tons of stuff" with "a lot" and "throw away" with "dispose". Using the recursive autoencoder, we can not only capture the word paraphrase similarity, but also the phrase similarity.

3.3. Dynamic Pooling and Paraphrase Classification

Given the RAE-derived representation of the sentence, we would like to calculate the similarity of two sentences. To deal with the arbitrary length of the sentence, RAE word representations are normalized into a fixed length vector with an algorithm called dynamic pooling. Every sentence fed into the RAE forms a binary tree representation. Given this, we can define a matrix M, where the rows and columns in this matrix represent two sentences with the different lengths i and j. Because this matrix includes all the non-terminal nodes and leaves in the binary tree, the matrix M's size is $2i-1 \times 2j-1$.

The dynamic pooling algorithm takes a matrix M as an input and turns it into matrix M' with the fixed size $n \times n$. This algorithm will divide the matrix M into n roughly equal parts. Every minimal value in the rectangular window is selected to form a $n \times n$ grid.

Given this $n \times n$ grid, we then classify each utterance as similar or not using a softmax classifier layer. The softmax classifier takes the matrix M' as an input, and outputs a confidence score that decided whether a user input and dialog database is a paraphrase. In our study, we also use this confidence score as the retrieval score when performing the RNN retrieval.

Table 1 shows how the NN-based retrieval captures the correlation between user input and the example database. The matrix shows the dynamic pooling layer of the two sentences. Similar sentence pairs have a clear diagonal of dark line, indicating low Euclidean distance. The softmax layer can then identify this pattern as a close or paraphrased sentence to the input query.

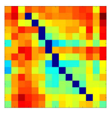
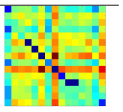
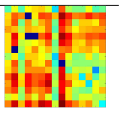
sim	Sentences	Matrix
0.94	S_1) Captain, we can not keep going fast on these icy roads. S_2) We can not keep going fast on these icy roads!	
0.60	S_1) Hold your fire! He's got a girl. S_2) Looks like he's got a hostage.	
0.38	S_1) Yes, I can see that too and I don't think it's so terrible. S_2) That's why I do all the thinking.	

Table 1. Sentence pairs.

4. EXPERIMENTAL SETUP

4.1. EBDM Setup

In this work, we use movie scripts to build our dialog corpus. We collect our movie script dialogues based on Friends TV show scripts[1], The Internet Movie Script Database[2], and The Daily Script[3]. The total number of gathered movie scripts is

[1]http://ufwebsite.tripod.com/scripts/scripts.htm
[2]http://imsdb.com/
[3]http://dailyscript.com/

1,786 with 1,042,288 dialog pairs. More details on the data can be found in [18]

We use natural language processing tools and Wordnet synsets provided by the NLTK toolkit[4] to perform the filtering on the EBDM example database. In general, we define Q_{test} as a query from the test set. To get the appropriate response similar to the actual response R_{test}, we use two retrieval techniques. TF-IDF based cosine similarity retrieval $cos_{sim}(Q_{test}, Q_{train})$ and NN-based retrieval $rnn(Q_{test}, Q_{train})$ are used to obtain the closest query Q_{train} in the train database, and the response R_{train} that trails the closest query is given as a response. We use Apache Lucene[5] to calculate the TF-IDF based cosine similarity, and calculate the similarity of the proposed method as described in the following section.

4.2. NN-based Retrieval Setup

In our experiment, we use the RAE trained with 150,000 sentences from NYT and AP section of the Gigaword corpus provided by Socher et al. [16]. To generate all the parse trees for the RAE algorithm, we use the Stanford parser [21]. We also employ the 100-dimensional word representations computed and provided by Turian et al. [22].

After performing pre-processing, filtering, and choosing dialog pairs that can be transformed into a vector of word representations, we finally use 10,033 dialog pairs as our training and test data. During the experiment, we separate our dialog pair data into 1,000 dialog pairs for test and 9,033 dialog pairs for train randomly.

To provide a balanced amount of similar and not similar queries during training, we do the cross product all training dialogues (9,033 pairs) with each other and calculate the syntactic-semantic similarity [18]

$$sim(S_1, S_2) = \alpha[sem_{sim}(S_1, S_2)] + (1-\alpha)[cos_{sim}(S_1, S_2)].$$

We assume that a similar query is obtained when the syntactic-semantic score is exclusively between 0.7 and 0.9, and a non-similar query is obtained when the syntactic-semantic score is exclusively between 0.2 and 0.4[6]. In the end, we obtained 1,421,338 pairs of training data with the ratio between similar and non-similar sentences being 50:50.

5. EVALUATION

We evaluate the system response objectively by calculating the system output response R_{output} similarity with the actual expected output R_{test}, and subjectively by asking the opinion of dialog system users.

5.1. Objective Evaluation

In the objective evaluation, we compare the NN-based retrieval (RNN) to the baseline system using TF-IDF cosine similarity retrieval (CSM). Figure 3 depicts the evaluation results. We evaluate the system performance objectively using the same scoring model (RNN and CSM) that we use for retrieving the system response. In order to obtain the evaluation score, these models calculate the similarity score between R_{output} and R_{test}.

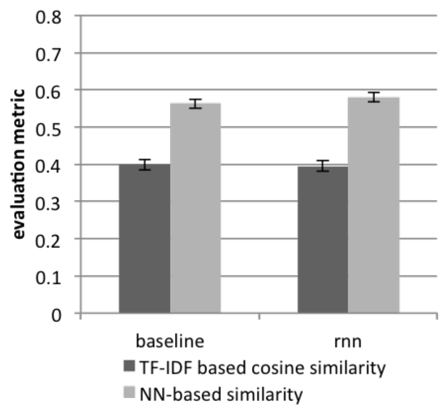


Fig. 3. Objective evaluation results.

From this graph, we can see that the two methods obtain roughly the same results. In order to look more closely at the evaluation results, we split all testing data into three cases (Table 2) based on each system's retrieval score.

1 Case 1 (closest example found; CEF) when the CSM retrieval score is more than a threshold. During this case, the CSM has managed to find a close example in the database.

2 Case 2 (out of vocabulary; OOV) when the CSM retrieval score is less than or equal to threshold and RNN retrieval score is more than the threshold. In this group, the CSM does not manage to find any close examples in the database, but the paraphrase identification model did.

3 Case 3 (out of vocabulary - non paraphrase; OOV-NP) when both the CSM and RNN retrieval scores are less than or equal to the threshold. For this case, there is no match with the given input for either method.

To determine the threshold value, we first measured the objective evaluation score at different thresholds as can be

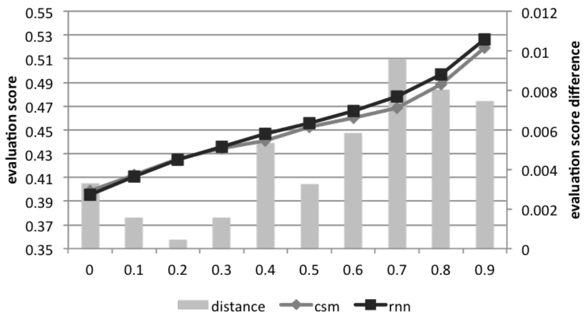
[4]http://nltk.org

[5]http://lucene.apache.org/

[6]Note that it would be better to manually create a corpus of similar and non-similar utterances, but this is extremely time consuming, so we take the more light-weight automatic approach in this paper.

	CSM > threshold	RNN > threshold
CEF	O	
OOV	X	O
OOV-NP	X	X

Table 2. Evaluation case.



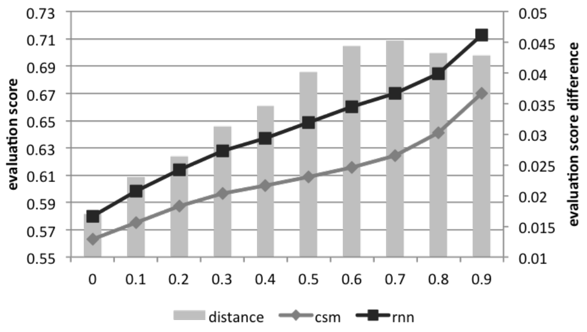


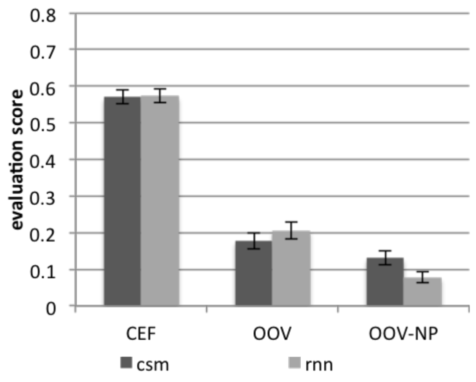
Fig. 4. Objective evaluation results for each threshold over TF-IDF based cosine similarity metrics (upper) and NN-based similarity metrics (lower).

seen in the Figure 4. We obtain a threshold score 0.7 by observing the longest evaluation score difference between RNN and CSM. Given this threshold, the amount of data in CEF, OOV, and OOV-NP respectively are 587, 206, and 207.

The details of the objective evaluation results for each of these groups is shown in Figure 5. The upper figure shows evaluation calculated by TF-IDF based cosine similarity (*cos-tf-idf*), the bottom figure shows evaluation calculated by NN-based similarity (*paraphrase*). Furthermore, by using the best threshold score in Figure 5, we can see that the NN-based retrieval RNN approach performs better compared to the baseline method in the OOV data, but the results is dropped in the case OOV-NP data.

5.2. Subjective Evaluation

In this evaluation, 10 human annotators were asked to give a naturalness score between 1-5 to the system response R_{output}



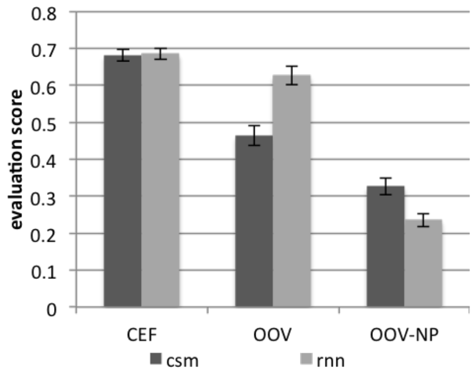


Fig. 5. Objective evaluation results. The upper figure shows evaluation over TF-IDF based cosine similarity, the lower figure shows evaluation over NN-based similarity.

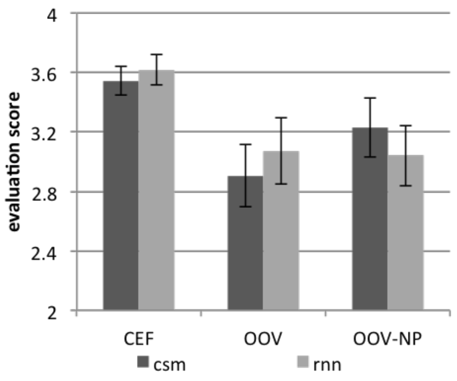


Fig. 6. Subjective evaluation results.

given the input query Q_{test}. A higher score indicates that the system is giving a natural and relevant system response to the user input, otherwise the lower scores indicate that the system doesn't give a related or natural response. Each person assesses 50 randomly selected query-response pairs that were evenly distributed over all systems. The results of this evaluation are shown in Figure 6. This evaluation shows that the RNN perform slightly better compared to the baseline approach for the CEF and OOV cases. However, when both of the systems can not find any good response (OOV-NP), the RNN response may not be related to the user utterance topic. In this situation, the user tends to choose the baseline approach response over the RNN response.

6. CONCLUSION

In this paper, we investigated recursive neural network paraphrase identification techniques to retrieve responses in a data-driven chat-oriented dialog system. We compare our current research with our previous work in TF-IDF based cosine similarity retrieval. An objective evaluation shows that the NN-based retrieval approach performs slightly better compared to the TF-IDF based cosine similarity retrieval approach when both methods find a response, or when only the RNN method finds a response. However this behavior is not shown in the OOV-NP case. As future work, applying the neural network word representations in the construction of the dialog example database can be a promising future direction.

7. ACKNOWLEDGMENT

Part of this work was supported by JSPS KAKENHI Grant Number 26870371.

8. REFERENCES

[1] S. Jung, C. Lee, and G.G. Lee, "Dialog studio: An example based spoken dialog system development workbench," in *Proc. of the Dialogs on dialog: Multidisciplinary Evaluation of Advanced Speech-based Interactive Systems. Interspeech2006-ICSLP satellite workshop,*, Pittsburgh, USA, 2006.

[2] C. Lee, S. Lee, S. Jung, K. Kim, D. Lee, and G.G. Lee, "Correlation-based query relaxation for example-based dialog modeling," in *Proc. of ASRU*, Merano, Italy, 2009, pp. 474–478.

[3] K. Kim, C. Lee, D. Lee, J. Choi, S. Jung, and G.G. Lee, "Modeling confirmations for example-based dialog management," in *Proc. of SLT*, Berkeley, California, USA, 2010, pp. 324–329.

[4] A. Ritter, C. Cherry, and W. B. Dolan, "Data-driven response generation in social media," in *Proc. of EMNLP*, Edinburgh, Scotland, UK., July 2011, pp. 583–593.

[5] J. Weizenbaum, "Eliza - a computer program for the study of natural language communication between man and machine," *Commun. ACM*, vol. 9, no. 1, pp. 36–45, 1966.

[6] R.Wallace, *Be Your Own Botmaster*, A.L.I.C.E A.I. Foundation, 2003.

[7] H. Murao, N. Kawaguchi., S. Matsubara, Y. Yamaguchi, and Y. Inagaki, "Example-based spoken dialogue system using WOZ system log," in *Proc. of SIGDIAL*, Saporo, Japan, 2003, pp. 140–148.

[8] F. Bessho, T. Harada, and Y. Kuniyoshi, "Dialog system using real-time crowdsourcing and twitter large-scale corpus," in *Proc. of SIGDIAL*, Seoul, South Korea, 2012, pp. 227–231.

[9] C. Lee, S. Jung, S. Kim, and G. G. Lee, "Example-based dialog modeling for practical multi-domain dialog system," *Speech Commun.*, vol. 51, no. 5, pp. 466–484, May 2009.

[10] N. Chambers and J. Allen, "Stochastic language generation in a dialogue system: Toward a domain independent generator.," in *Proc. of SIGDIAL*, Cambridge, Massachusetts, USA, 2004, pp. 9–18.

[11] R. E. Banchs and H. Li, "IRIS: a chat-oriented dialogue system based on the vector space model," in *Proc. of ACL (System Demonstrations)*, 2012, pp. 37–42.

[12] R. E. Banchs, "Movie-dic: a movie dialogue corpus for research and development.," in *Proc. of ACL (2)*, 2012, pp. 203–207.

[13] L. Nio, S. Sakti, G. Neubig, T. Toda, and S. Nakamura, "Combination of example-based and smt-based approaches in a chat-oriented dialog system," in *Proc. of ICE-ID*, 2013.

[14] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. of ICML*, New York, NY, USA, 2008, pp. 160–167.

[15] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, "Semantic compositionality through recursive matrix-vector spaces," in *Proc. of EMNLP*, 2012.

[16] R. Socher, E. H. Huang, J. Pennington, A. Y. Ng, and C. D. Manning, "Dynamic pooling and unfolding recursive autoencoders for paraphrase detection," in *Advances in Neural Information Processing Systems 24*. 2011.

[17] L. Nio, S. Sakti, G. Neubig, T. Toda, M. Adriani, and S. Nakamura, "Developing non-goal dialog system based on examples of drama television," in *Proc. of IWSDS*, Paris, France, 12 2012.

[18] L. Nio, S. Sakti, G. Neubig, T. Toda, and S. Nakamura, "Utilizing human-to-human conversation examples for a multi domain chat-oriented dialog system," *IEICE Transactions on Information and Systems*, June 2014.

[19] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Commun. ACM*, vol. 18, no. 11, pp. 613–620, Nov. 1975.

[20] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *J. Mach. Learn. Res.*, vol. 3, pp. 1137–1155, Mar.

[21] D. Klein and C. D. Manning, "Accurate unlexicalized parsing," in *Proc. of ACL*, Stroudsburg, PA, USA, 2003, pp. 423–430.

[22] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: A simple and general method for semi-supervised learning," in *Proc. of ACL*, 2010.