PAPER Special Section on Recent Advances in Machine Learning for Spoken Language Processing

Neural Network Approaches to Dialog Response Retrieval and Generation

Lasguido NIO[†a)], **Sakriani SAKTI**[†b)], **Graham NEUBIG**[†c)], **Koichiro YOSHINO**[†d)], ***Nonmembers***, ***and*** **Satoshi NAKAMURA**[†e)], ***Member***

SUMMARY In this work, we propose a new statistical model for building robust dialog systems using neural networks to either retrieve or generate dialog response based on an existing data sources. In the retrieval task, we propose an approach that uses paraphrase identification during the retrieval process. This is done by employing recursive autoencoders and dynamic pooling to determine whether two sentences with arbitrary length have the same meaning. For both the generation and retrieval tasks, we propose a model using long short term memory (LSTM) neural networks that works by first using an LSTM encoder to read in the user's utterance into a continuous vector-space representation, then using an LSTM decoder to generate the most probable word sequence. An evaluation based on objective and subjective metrics shows that the new proposed approaches have the ability to deal with user inputs that are not well covered in the database compared to standard example-based dialog baselines.

key words: *example-based dialog system, dialog system, response retrieval, response generation, long short term memory neural network*

1. Introduction

Natural language dialogue systems promise to establish efficient interfaces for communication between humans and computers [1]–[5]. One way to create a simple yet effective dialog system is using example-based dialog modeling (EBDM) [6]–[9]. EBDM is a data-driven approach for creating dialog systems that choose how to respond to user input based on a large database of examples consisting of an utterance, and a corresponding natural reply to that utterance. Given a user input, the system then performs response *retrieval*, selecting the highest scoring response from the existing utterances in the database. EBDM presents a lightweight alternative to more conventional methods for constructing dialog systems, as it only requires the construction of an example base, and has also been shown effective in a number of dialog scenarios. In particular, this approach is able to generate highly natural output when an example that matches closely with the user query is included in the database and the example is appropriately retrieved [6]–[8], [10]–[13].

Manuscript received February 5, 2016.
Manuscript revised May 25, 2016.
Manuscript publicized July 19, 2016.

[†]The authors are with Nara Institute of Science and Technology, ●●●●●-shi, ●●●–●●●● Japan.
a) E-mail: lasguido@yahoo.com
b) E-mail: ssakti@is.naist.jp
c) E-mail: neubig@is.naist.jp
d) E-mail: koichiro@is.naist.jp
e) E-mail: s-nakamura@is.naist.jp

DOI: 10.1587/transinf.2016SLP0018

However, dealing with sparse human language and a finite query-response database, we can imagine easily that such system may fail when attempting to respond to a user utterance that does not match closely with one of the examples in the database. We define this kind of problem as an out of example (OOE) problem. One way to overcome this problem is using response *generation*. This approach uses the dialog examples as data to train a model that can generate responses not included in the database. Generation has the potential to be more robust to OOE user inputs, but also may generate responses that are incomprehensible to human users [13]. Generation models originally adapted statistical machine translation (SMT) to utilize a query-response dialog database as a parallel corpus to "translate" between query input and response output [9], [14]. In the recent developments, there have also been several alternative approaches developed simultaneously to this work that utilize neural networks for dialog generation [15]–[17].

In this work, we propose and compare methods for using neural networks for both dialog response retrieval and generation. The reason why we focus on neural networks specifically is that in dialog tasks, we are dealing with a tenuous and indirect relationship between a dialog utterance and responses [9]. Continuous distributed representations used by neural networks have proven useful for expressing these sorts of soft relationships in natural language [18], and thus may be more able to accommodate such soft associations.

Particularly, we make several contributions*:

- We propose a new EBDM method to retrieve dialog responses from the database by utilizing a neural-network based paraphrase matching algorithm. In this approach, we model the example in our dialog-pair database and the 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.
- We propose a method that utilizes LSTM neural networks to perform response retrieval, in addition to generating responses directly. Given the LSTM response generation model, we calculate the perplexity of each response in the database conditioned on the user query,

*Part of this work on neural-network based paraphrase matching have been published [19]. This paper add to the previous works by proposing a dialog system that utilize LSTM neural network

and obtain the best-scoring response candidate directly from the dialog database. This way we can reduce the chance of grammatical errors that occur when generating dialog responses, while using the ability of neural networks to perform soft matching to improve retrieval accuracy.

- We perform an analysis and contrastive experiment to compare our proposed approach with the state-of-the-art baseline approaches in data-driven chat-oriented dialog. We evaluate using both automatic and manual evaluation, over examples that are well covered by the example base, as well as examples for which a close match does not exist.

2. Related Works

Earlier efforts to incorporate a data-driven approach into dialog systems rely on two main approaches. The first is *response retrieval* approaches that search for the most appropriate response in a conversation database [6]–[8], [10], [11], [13], [19]. However, in the case that no response in the database could adequately respond to a given utterance, this approach will fail. *Response generation* [15]–[17] which has the ability to generate a new responses, is arguably robust in handling user input comparing to the other approach, however this approach sometimes generates unnatural responses that are incomprehensible to the user [9].

There have been a number of works on response generation for data-driven dialog systems. The first work in data-driven response generation utilized a statistical machine translation system to learning a patterns between queries and response in conversational data [9]. On the other hand, many recent works focus on models based on recurrent neural network language models (RNNLM) [20]. Sordoni et al. [15] employ an RNN architecture to generate responses from a social media corpus, and Vinyals et al. [16] present a long short-term memory (LSTM) neural network encoder-decoders to generate dialog responses using movie subtitles or IT support line chats. More recently Wen et al. [17] demonstrate a more advanced LSTM that able to control a response semantically by considering dialogue act feature.

Our LSTM response generation approach (which we developed simultaneously to earlier efforts) is similar, but also can be used to perform response retrieval. We hope that this will reduce our chance of grammatical errors that occur when generating a dialog response. Furthermore, our works include a contrastive experiment, comparing the new approaches to baseline methods in a data-driven dialog system.

3. Baseline Response Retrieval and Generation

In response retrieval and generation, we take user utterance Q as an input and choose a response R', which is obtained from the query-response dialog pairs in the database $\langle Q', R' \rangle \in D$. Methods to obtain response R' can be classified as (1) retrieval and (2) generation. Though different, these techniques both can be viewed as a scoring problem where we try to assign score $S(Q, R')$ that measures the appropriateness of response candidate R' given user utterance Q. In response retrieval, we find the R' with the highest score out of all the examples in the database D:

$$\hat{R} = \underset{R' \in D}{\operatorname{argmax}} S(Q, R'). \tag{1}$$

On the other hand, response generation finds hypothesis $R \approx R'$ with the highest score out of all the possible sentence hypotheses $\mathcal{R}$:

$$\hat{R} = \underset{R \in \mathcal{R}}{\operatorname{argmax}} S(Q, R). \tag{2}$$

There are many ways to retrieve or generate responses in EBDM systems. In this work, we implement (1) similarity-based retrieval [10], [11] as our baseline response retrieval approach, and (2) phrase-based statistical machine translation (SMT response generation) [9] as our baseline response generation approach.

3.1 Similarity-Based Retrieval

Our first baseline, **cosine similarity-based retrieval** (CSM), works by matching the user query input Q and all the possible queries Q' from the query-response pairs $\langle Q', R' \rangle$ in the dialog database. This method returns response R' that corresponded to the highest scoring Q'. In these models, we define a score $S(Q, Q')$ measuring similarity between user and database queries, and assume that $S(Q, R') \approx S(Q, Q')$. We define $S(Q, Q')$ as TF-IDF based cosine similarity. TF-IDF based cosine similarity calculates cosine similarity over the TF-IDF weighted term vector of two sentences, S_1 and S_2 [10]

$$cos(S_1, S_2) = \frac{S_1 \cdot S_2}{\| S_1 \| \| S_2 \|} \tag{3}$$

To calculate term vector S from query Q, we first create a vector consisting of the frequency of all word in S, then weight each word with its TF-IDF value [21], a measure of term importance widely used in information retrieval.

3.2 SMT Response Generation

Next, in the **SMT response generation** (SMT) approach, we treat dialog-pair data $\langle Q', R' \rangle$ as a parallel corpus for training an SMT system. Given the trained SMT system, the user dialog is treated as an input and "translated" into the system response. The system response is chosen to be system output $\hat{R}$ of maximal probability given the user input Q

$$\hat{R} = \underset{R \in \mathcal{R}}{\operatorname{argmax}} P(R \mid Q), \tag{4}$$

where $P(R \mid Q)$ is a probability distribution of candidate response R given the user input Q.

4. RAE Paraphrase-Based Retrieval

Simple methods such as cosine similarity have problems

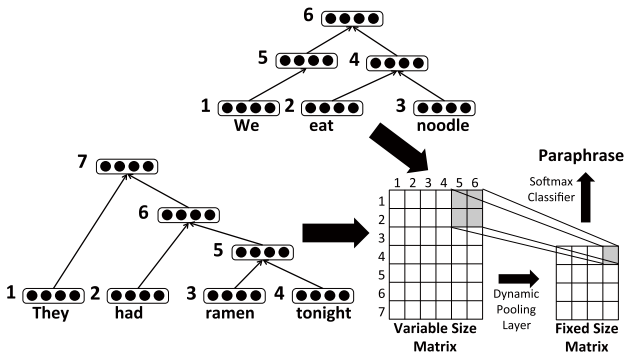


Fig. 1 Overview of neural-network-based retrieval.

with robustness [19]. Thus we need a more sophisticated approach to retrieve a response from the example database. 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 the example database using neural word representations, and passing it to the softmax classifier that calculates a probability that the user input Q and query in the example database Q' are paraphrases. Thus we can view the scoring function $S(Q, Q')$ as being this paraphrase probability.

Adopting the work of [22], we utilize recursive autoencoders (RAE), dynamic pooling, and a softmax classifier to decide whether the sentence is paraphrased or not. In the following sections we describe: (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 method is depicted in Fig. 1.

4.1 Word Representations

A distributed word representation is an n-dimensional vector of continuous values used to represent a word i in the vocabulary D ($i \in D$). They are often obtained by joint learning of neural network language models and distributed representation for words [23]. The reason why word representations are useful is that they allow for soft matching of similar words when exact matches are not available. This is useful especially when we are dealing with the large vocabularies. Without soft matching, the response generator has a tendency to fail and respond to the user input with another uncorrelated response based on superficial overlap of the words that do happen to have an exact match.

4.2 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 [19].

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.

4.3 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.

5. LSTM Response Retrieval and Generation

In this section, we explain about response generation and retrieval methods using LSTM recurrent neural networks. LSTMs have shown impressive ability to capture syntactic and semantic information of sentences in other applications [24]–[26], and it is reasonable that this will carry over to dialog as well.

5.1 Long Short Term Memory Model

We can view each query Q and response R' dialog pair as a set of a words $(q_1, \ldots, q_I)$ and $(r'_1, \ldots, r'_{I'})$. By doing so, we can formulate a conditional probability of the response given the query as $P(R'|Q) = P(r'_1, \ldots, r'_{I'}|q_1, \ldots, q_I)$.

Before the LSTM takes a word from the input sentence, we transform each word into a distributed word representation. A distributed word representation is an n-dimensional vector of continuous values used to represent a word in the vocabulary [18], [23]. Each word in the dictionary ($w \in W$) is embedded into n-dimensional space $L \in \mathbb{R}^{n \times |W|}$. From this

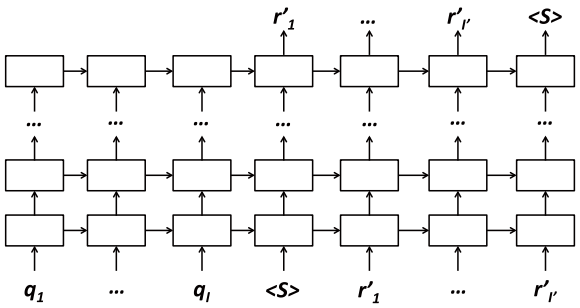


Fig. 2 LSTM neural model over time.

representation, a word vector can be seen as a single vector in the column L.

Each LSTM layer is composed of input gates i, forget gates f, output gates o, and memory cells c. Mathematically this can be viewed as:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i), \quad (5)$$

$$f_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_f), \quad (6)$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c), \quad (7)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o), \quad (8)$$

$$h_t = o_t \tanh(c_t), \quad (9)$$

where x_t is an input to the LSTM, in our case a single distributed word vector L of word w, σ is a logistic sigmoid function, and h is a hidden vector. The weight W and b subscript respectively represent the edge connection matrix and bias vector. For example W_{xc} indicates the input-cell (xc) weight matrix. To calculate input gates i we apply the logistic sigmoid over the sum of dot products of (1) input weight matrix W_{xi} and word input x_t, (2) hidden-input weight matrix W_{hi} and the previous hidden vector h_{t-1}, (3) cell-input weight matrix W_{ci} and previous memory cell c_{t-1}, and (4) input bias vector b_i.

At the end of each LSTM, we calculate the output probability by performing an affine transform on the LSTM output h_t, and calculating the probability with the softmax function:

$$P(r'_{t+1}|h) = softmax(W_{hy}h_t + b), \quad (10)$$

where W_{hy} is a hidden-output weight matrix, and b is a bias.

Our model consists of an LSTM encoder-decoder with two LSTMs, one for the query sequence and another for the response sequence. The details of how the LSTM works can be seen in the Fig. 2. Given the query $(q_1, \ldots, q_I)$ the network will predict a response $(r'_1, \ldots, r'_{I'})$ as the output. Note that the system starts to decode the output after reading the input and receiving the end of sentence symbol "<S>".

The conditional probability of the next word in the response sentence is calculated conditioned on a hidden representation h and memory cell that encodes the input query $q_1, \ldots, q_I$, and the previously generated words of the response sequence

$$P(r'_1, \ldots, r'_{I'}|q_1, \ldots, q_I) = \prod_{i=1}^{I'} P(r'_i|h). \quad (11)$$

We also experiment with deep LSTMs that stack memory cells one after another. During training we utilize back propagation through time [27] to calculate the gradient over the full sequence, minimizing the negative log likelihood using stochastic gradient descent. Using the development data, we calculate the LSTM loss function, and decrease the network learning rate by half when there is no improvement over time. The learning is terminated when the learning rate is lower than a threshold.

5.2 LSTM Response Generation

As explained above, to encode the input sentence, we feed it word by word $(q_1, \ldots, q_I)$ to the LSTM model. By following the word probability $P(r'|h)$ from Eq. (11), we calculate the word with the highest probability as follows:

$$\hat{R} = \sum \underset{r' \in \mathsf{R}}{\operatorname{argmax}} P(r'|h). \quad (12)$$

After encoding the target sentence, the LSTM decoder is used to generate output word by word. We search for the most likely response by using a left-to-right beam search decoder which maintains a small amount number h of partial hypotheses at each time step, and discard the rest [25]. When we reach a symbol "<S>" and append it to the highest-scoring hypothesis, we end the search.

5.3 LSTM Response Retrieval

Different from the LSTM response generation, in LSTM response retrieval we calculate $P(R'|Q)$ for every response candidate R' in the dialog database based on its conditional probability $\log P(R'|Q)$ divided by the number of words $|R'|$.

$$\hat{R} = \underset{R' \in D}{\operatorname{argmax}} \frac{\log P(R'|Q)}{|R'|}. \quad (13)$$

This score tells us how likely a response candidate is to be an output response, given the user utterance sentence and the LSTM model. We use this score to retrieve the highest scoring response from the dialog database.

6. Construction of Dialog Corpora

The dialogue corpora that we constructed in this study is based on dialog-pair sentences that have been extracted from movie script. Our method to construct a dialog corpus can be briefly explained as three main steps: (1) preprocessing, which removes unnecessary information and normalizes the text, (2) dialog pair extraction, which ensures that the conversation is between two people talking with each other, and (3) semantic similarity calculation, which ensures that the each query-response pair is semantically related. We describe each step briefly in the following sections. More discussion about data collection, processing, and the advantage

Actor	Correlated tri-turn
C	He's great, but I'm not attracted to him at all!
A	Still?!
C	Yeah! It's driving me crazy.
Actor	**Un-correlated tri-turn**
B	Hi, where were you yesterday?
A	How'd the date go with Mr. Millionaire?
B	Who is Mr. Millionaire?

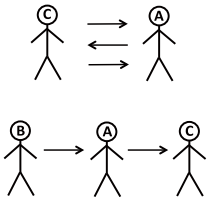


Fig. 3 Example of a tri-turn with two actors.

of using tri-turn filtering can be obtained in [10].

6.1 Data Collection and Preprocessing

We collect the data by creating a script to automatically download web pages that contain a movie script. Preprocessing of the movie scripts is done by transforming raw HTML files into easily readable text format. Since we use a variety of sources of movie scripts that had various formats, we implemented several parsing algorithms to fetch the information from the raw movie conversation. Furthermore, unnecessary explanatory information about the movie scenes is also removed.

6.2 Dialog Turn Extraction

To ensure that the dialog example database contains only query-response pairs, we propose a simple and intuitive method for selection of the dialog data: trigram turn sequences, or *tri-turns*. A tri-turn is defined as three turns in a conversation between two actors A and B that has the pattern A-B-A. In other words, within a tri-turn the first and last dialog turn are performed by the same actor and the second dialog turn is performed by the other actor.

We found that when we observed this pattern, in the great majority of the cases this indicated that the first and second utterances, as well as the second and third utterances, formed a proper input-response pair as shown in the C-A-C tri-turn in Fig. 3. However, noisy cases which contain uncorrelated turns still exist (see the B-A-B tri-turn in Fig. 3), this happens because the speakers are not actually speaking to each-other. To address this problem, we perform further filtering using the semantic similarity measure described in the following section.

6.3 Semantic Similarity

Semantic similarity [28], shown in Eq. (14), is used to ensure a strong semantic relationship between each dialog turn in the dialog-pair data, by computing the similarity between WordNet[†] synsets in each dialog turn. The dialog pairs with high similarity are then extracted and included into database.

[†]http://wordnet.princeton.edu/

$$sem_{sim}(S_1, S_2) = \frac{2 \times |S_{syn1} \cap S_{syn2}|}{|S_{syn1}| + |S_{syn2}|} \tag{14}$$

7. Experimental Setup

7.1 Data Preparation

As data for our experiments, we use movie script data collected from Friends TV show scripts[††], The Internet Movie Script Database[†††], and The Daily Script[††††]. In total we gather 1,786 movie scripts, and extract 1,042,288 dialog pair candidates. In the end, we use 10,033 dialog pairs and separate them into 9,033 dialog pairs for training data and 1,000 dialog pairs for testing data.

As mentioned in the introduction, the effectiveness of example-based dialog largely depends on whether a close example exists in the database. To examine how well each method works when a close example exists or doesn't exist, we further divide the test dialog pair data into two cases [19]:

1 Close example found (CEF) - (587 examples): A given user query is available or there exists a close example in the dialog database. This happens when baseline CSM retrieval score is more than a threshold 0.7 [19].
2 Out of example (OOE) - (413 examples): The rest of the queries under the threshold.

7.2 Paraphrased-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. [22]. To generate all the parse trees for the RAE algorithm, we use the Stanford parser [29]. We also employ the 100-dimensional word representations computed and provided by Turian et al. [30].

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 [10] $sim(S_1, S_2) = \alpha[sem_{sim}(S_1, S_2)] + (1-\alpha)[cos_{sim}(S_1, S_2)]$. 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[†††††]. In the end, we obtained 1,421,338 pairs of training data with the ratio between similar and non-similar sentences being 50:50.

7.3 LSTM Network Setup

In order to train our LSTM network, we separate our dia-

[††]http://ufwebsite.tripod.com/scripts/scripts.htm
[†††]http://imsdb.com/
[††††]http://dailyscript.com/
[†††††]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.

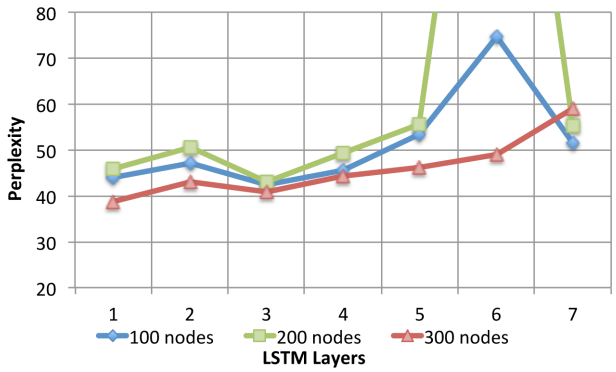


Fig. 4 LSTM model perplexity.

log pair training data into 7,227 and 1,806 examples[†] for training and development sets. During training, we used the training set to learn the parameters, and development set as a criterion to evaluate the network performance and decide whether to continue training or not.

Before evaluating the LSTM-based methods on actual dialog performance, we first evaluate the perplexity of the model on the development set for various numbers of nodes in the hidden layers (100, 200, 300), and various numbers of hidden layers (1-7).

The perplexity results of the network training can be seen in Fig. 4. The best performance of the various settings is achieved by the 1 layer LSTM with 300 nodes in the hidden layer, with a perplexity score of 38.73. We use this network in our dialog-based evaluation in the following sections.

8. Evaluation

In this work, we evaluate the system responses objectively by calculating the system output response $\hat{R}$ similarity with the expected output R, and subjectively by performing a questionnaire with actual users.

8.1 Objective Evaluation

In the objective evaluation, we calculate the similarity between the system response $\hat{R}$ and the expected output R with (1) TF-IDF cosine similarity, which focuses on content word similarity, and (2) BLEU-4, which focues on fluency and local word order [31]. We compare our baseline retrieval systems (CSM) with the proposed paraphrase-based response retrieval (PARA) and LSTM response retrieval (LSTM-RET), and baseline response generation system (SMT) with LSTM response generation approaches (LSTM-GEN).

The result of the objective evaluation over the cosine TF-IDF similarity metrics can be seen in the top section of Fig. 5. This objective evaluation shows that both LSTM-RET and LSTM-GEN approaches significantly outperform the baselines not only in the OOE, but also in the CEF case. In this figure, we can also see that SMT approach can pick a good selection of words during generating a response in OOE case, however in most of the cases we observed that these responses are incomprehensible [10]. This behavior resulted in the lower performance of subjective evaluation, as seen in the following section.

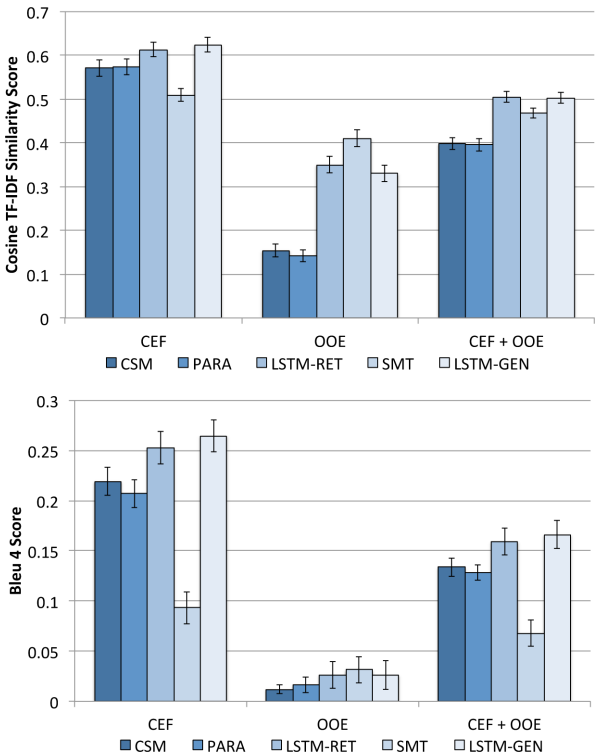


Fig. 5 Objective evaluation results over the cosine TF-IDF similarity (top) and BLEU-4 (bottom) metric.

Next, we evaluate our system performance with the BLEU-4 metric to calculate how well the response can capture response fluency. The results of the BLEU-4 evaluation can be seen in the bottom section of Fig. 5. From this evaluation, we can see that again both LSTM approaches perform better, especially in the CEF case.

Though the PARA performance is on par with the baseline overall, we can see that it is slightly better than the baseline in the OOE case. This is because PARA approach performs retrieval capturing paraphrase features. An example of how this approach captures paraphrase features can be seen in Sect. 8.3.

8.2 Subjective Evaluation

Next, we report two varieties of subjective evaluation: naturalness and relevance. A response is categorized as natural if the sentence is comprehensible and likely to be generated by a human. On the other hand, we define a relevant response as a response that is appropriately related to the user query sentence. We asked 5 human annotators to give a score be-

[†]This data is relatively small, but the limit of what we could collect after semantic similarity filtering. Test with larger data are reserved for future work.

Table 1 Various responses for each dialog system.

	CEF	OOE
User Utterance	What do you mean by that?	Do you know why I'm here?
CSM	I mean you're different.	I know what you mean.
PARA	I do different things on different days.	If you know what I am, you know very well I can wipe you from existence.
LSTM-GEN	I mean, what do you mean?	I know you are.
SMT	What do I mean by that.	I know what I am.
LSTM-RET	I want to talk about me.	I know. I was here.

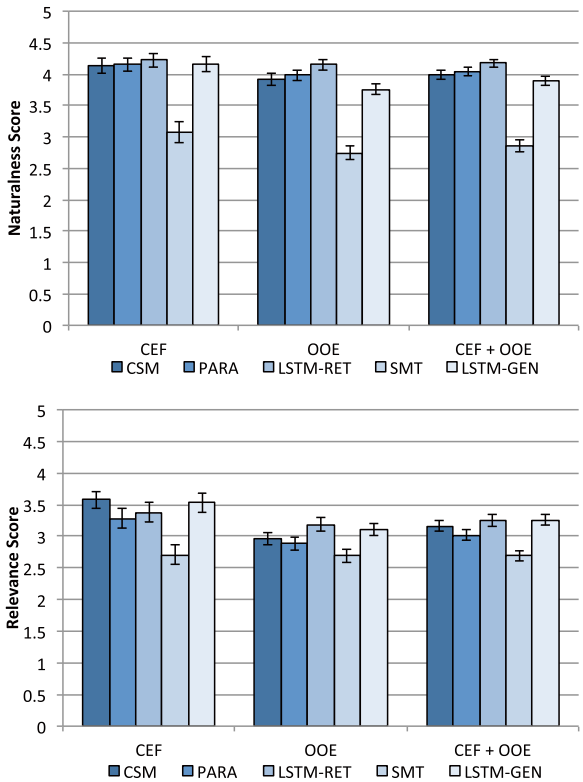


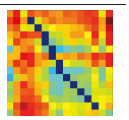
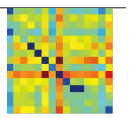
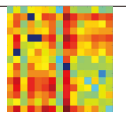
Fig. 6 Naturalness (top) and Relevance (bottom) for each system.

tween 1-5 to the system response. Each person was asked to annotate 255 randomly selected query-response pairs that were evenly distributed over all the systems.

The subjective evaluation on naturalness shows that our proposed methods PARA, LSTM-RET, and LSTM-GEN give responses that are on par with the baseline approaches (see Fig. 6 at the top section), with LSTM-RET performing slightly better compared to the baseline approaches in both the CEF and OOE cases. By observing the LSTM-GEN generated responses sentence by sentence, we found that most of the responses are short, compared to the retrieved responses, which sometimes generate long responses. These short responses are easy to comprehend and reduce the chance of grammatical mistakes.

By looking at the subjective relevance evaluation in the bottom section of Fig. 6, we can observe that the PARA relevance score is slightly under the baseline. It is tell us that though PARA approach is manage to captures the paraphrase

Table 2 This table shows a correlation between two sentences, user input and example database. We calculate syntactic-semantic score *sim* [10] for each utterance pair ($S1$ and $S2$).

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.	

features, it is still difficult to give a relevant answer to the user query. On the other hand, the LSTM-RET retrieval and LSTM-GEN generation approaches, compared to the other approaches, can perform significantly better in the OOE case. Furthemore, the performance of the LSTM-RET retrieval and LSTM-GEN generation is almost the same for the CEF case. This indicates that the LSTM response generation is relatively robust, even in cases where a close match does not exist in the database.

Finally, we show some results from each of the systems in Table 1. First, we can see that all algorithms performs relatively well on the CEF case, where we can find a good example that matches the user utterance in the dialog database. In the OOE case, both CSM and PARA may give an uncorrelated response, as they might not find a good response in the dialog database. On the other hand, both LSTM-RET and LSTM-GEN are more likely to give a short, meaningful, and correlated response, which corroborates the results of [16]. In many cases, we found that LSTM-GEN and LSTM-RET give a similar response, which happens because the same LSTM neural network model is used for both systems.

8.3 Paraphrase-Based Retrieval Performance

Table 2 shows how the paraphrased-based response retrieval (PARA) 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

Table 3 Comparison with retrieval based distributed representations.

	BLEU-4 Score		Cos TF-IDF Score	
	CEF	OOE	CEF	OOE
CSM	0.2191	0.0121	0.5706	0.1541
CSM-EMBD	0.2625	0.0251	0.6548	0.3035
LSTM-RET	0.2526	0.0261	0.6130	0.3498
LSTM-GEN	0.2646	0.0261	0.6240	0.3302

close or paraphrased sentence to the input query.

8.4 Comparison with Retrieval Based on Distributed Representations

One difference between the cosine TF-IDF retrieval approach (CSM) and response generation with LSTM (LSTM-GEN) is that LSTM-GEN employs distributed word embeddings while CSM employs discrete representations for words. To examine the effect of discrete vs. distributed representations, we perform a follow-up study on retrieving responses with the cosine similarity over a vector of word embeddings (CSM-EMBD). The result of our experiment can be seen in the Table 3. While both the performance of LSTM-GEN and LSTM-RET is on par with the CSM-EMBD, we can see that by employing the distributed word embedding (CSM-EMBD, LSTM-RET, LSTM-GEN) these approaches could surpass CSM approach that does not use distributed word embeddings. Furthermore we could also see both LSTM-GEN and LSTM-RET are slightly better compared to the CSM-EMBD in the OOE case.

In addition, the LSTM-GEN approach has an advantage in that it is more efficient in terms of the computational complexity of creating a response. In normal response retrieval (CSM-EMBD) we need to traverse all dialog data in the database, and thus complexity is O(n) where n is the amount of data in the dialog database. On the other hand, LSTM-GEN has a complexity of O(1) in the data size because we don't have to traverse all the data. Knowing this is useful especially when we want to deliver this dialog framework to the end user in real time.

9. Conclusion

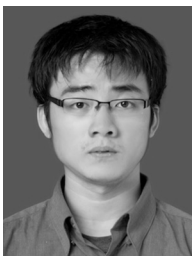
In this work, we investigate several approaches to create a robust data-driven dialog agent by proposing a new EBDM method to retrieve and generate a dialog response from the database utilizing a neural network model. Our experimental evaluation shows that these neural network retrieval and generation approaches were effective, and can generate a response that on par with the baseline system even better. Furthermore, by focusing on addressing the case where a similar example does not exists in the training data (OOE case), we found out that our proposed approach can perform well, improving the robustness over the baseline approaches.

There are still many improvement can be done to these methods. One future possibility is to incorporate a knowledge base to the dialog agent. Another way is to maintain long-term consistency controlling conversation context.

References

[1] J. Holmes and W. Holmes, Speech Synthesis and Recognition, 2nd ed., Taylor & Francis, Inc., Bristol, PA, USA, 2002.
[2] S. Seneff, L. Hirschman, and V. Zue, "Interactive problem solving and dialogue in the ATIS domain," Proc. Fourth DARPA Speech and Natural Language Workshop, pp.354–359, 1991.
[3] R. Wallace, Be Your Own Botmaster, A.L.I.C.E A.I. Foundation, 2003.
[4] M. Walker, J. Aberdeen, J. Boland, E. Bratt, J. Garofolo, L. Hirschman, A. Le, S. Lee, S. Narayanan, K. Papineni, B. Pellom, J. Polifroni, A. Potamianos, P. Prabhu, A. Rudnicky, G. Sanders, S. Seneff, D. Stallard, and S. Whittaker, "DARPA communicator dialog travel planning systems: the June 2000 data collection," Proc. EUROSPEECH, pp.1371–1374, 2000.
[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] S. Jung, C. Lee, and G. Lee, "Dialog studio: An example based spoken dialog system development workbench," Proc. Dialogs on dialog: Multidisciplinary Evaluation of Advanced Speech-based Interactive Systems. Interspeech2006-ICSLP satellite workshop, Pittsburgh, USA, 2006.
[7] C. Lee, S. Lee, S. Jung, K. Kim, D. Lee, and G.G. Lee, "Correlation-based query relaxation for example-based dialog modeling," Proc. ASRU, Merano, Italy, pp.474–478, 2009.
[8] K. Kim, C. Lee, D. Lee, J. Choi, S. Jung, and G.G. Lee, "Modeling confirmations for example-based dialog management," Proc. SLT, Berkeley, California, USA, pp.324–329, 2010.
[9] A. Ritter, C. Cherry, and W.B. Dolan, "Data-driven response generation in social media," Proc. EMNLP, Edinburgh, Scotland, UK., pp.583–593, July 2011.
[10] 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 Trans. Inf. & Syst., vol.E97-D, no.6, pp.1497–1505, June 2014.
[11] R.E. Banchs and H. Li, "IRIS: a chat-oriented dialogue system based on the vector space model," Proc. ACL (System Demonstrations), pp.37–42, 2012.
[12] R.E. Banchs, "Movie-dic: a movie dialogue corpus for research and development.," Proc. ACL (2), pp.203–207, 2012.
[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," Proc. ICE-ID, 2013.
[14] A. Shin, R. Sasano, H. Takamura, and M. Okumura, "Context-dependent automatic response generation using statistical machine translation techniques," Proc. NAACL-HLT, pp.1345–1350, 2015.
[15] 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," Proc. NAACL-HLT, pp.196–205, 2015.
[16] O. Vinyals and Q. Le, "A neural conversational model," Proc. ICML Deep Learning Workshop, Lille, France, 2015.
[17] T.-H. Wen, M. Gasic, N. Mrkšić, P.-H. Su, D. Vandyke, and S. Young, "Semantically conditioned LSTM-based natural language generation for spoken dialogue systems," Proc. EMNLP, Lisbon, Portugal, pp.1711–1721, 2015.
[18] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," Proc. ICML, Helsinki, Finland, pp.160–167, 2008.
[19] L. Nio, S. Sakti, G. Neubig, T. Toda, and S. Nakamura, "Improving the robustness of example-based dialog retrieval using recursive neural network paraphrase identification," Proc. IEEE SLT, pp.306–311, 2014.
[20] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, "Recurrent neural network based language model,"

Proc. INTERSPEECH, Chiba, Japan, 2010.
[21] 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.
[22] 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, Curran Associates, Inc., 2011.
[23] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," J. Mach. Learn. Res., vol.3, pp.1137–1155, 2003.
[24] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol.9, no.8, pp.1735–1780, 1997.
[25] I. Sutskever, O. Vinyals, and Q. Le, "Sequence to sequence learning with neural networks," Proc. NIPS, 2014.
[26] M. Auli, M. Galley, C. Quirk, and G. Zweig, "Joint language and translation modeling with recurrent neural networks," Proc. EMNLP, Seattle, Washington, USA, 2013.
[27] P.J. Werbos, "Backpropagation through time: what does it do and how to do it," Proc. IEEE, vol.78, no.10, pp.1550–1560, 1990.
[28] D. Liu, Z. Liu, and Q. Dong, "A dependency grammar and wordnet based sentence similarity measure," Journal of Computational Information Systems, vol.8, no.3, pp.1027–1035, 2012.
[29] D. Klein and C.D. Manning, "Accurate unlexicalized parsing," Proc. ACL, Stroudsburg, Pennsylvania, USA, pp.423–430, 2003.
[30] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: A simple and general method for semi-supervised learning," Proc. ACL, 2010.
[31] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: A method for automatic evaluation of machine translation," Proc. ACL, Philadelphia, Pennsylvania, USA, pp.311–318, 2002.



Lasguido Nio received his B.C.S. and his M.C.S. degree from Universitas Indonesia, Indonesia in 2012 and 2013. He is currently a Doctor Student in Nara Institute of Science and Technology, Nara, Japan. His research interest include information retrieval and dialog systems.

Sakriani Sakti received her B.E. degree in Informatics (cum laude) from Bandung Institute of Technology, Indonesia, in 1999. In 2000, she received "DAAD-Siemens Program Asia 21st Century" Award to study in Communication Technology, University of Ulm, Germany, and received her MSc degree in 2002. During her thesis work, she worked with Speech Understanding Department, DaimlerChrysler Research Center, Ulm, Germany. Between 2003-2009, she worked as a researcher at ATR SLC Labs, Japan, and during 2006-2011, she worked as an expert researcher at NICT SLC Groups, Japan. While working with ATR-NICT, Japan, she continued her study (2005-2008) with Dialog Systems Group University of Ulm, Germany, and received her PhD degree in 2008. She actively involved in collaboration activities such as Asian Pacific Telecommunity Project (2003-2007), A-STAR and U-STAR (2006-2011). She also served as a visiting professor of Computer Science Department, University of Indonesia (UI) in 2009-2011. Currently, she is an assistant professor of the Augmented Human Communication Lab, NAIST, Japan. She is a member of JNS, SFN, ASJ, ISCA, IEICE amd IEEE. Her research interests include statistical pattern recognition, speech recognition, spoken language translation, cognitive communication, and graphical modeling framework.

Graham Neubig received his B.E. from University of Illinois, Urbana-Champaign, U.S.A, in 2005, and his M.E. and Ph.D. in informatics from Kyoto University, Kyoto, Japan in 2010 and 2012 respectively. He is currently an assistant professor at the Nara Institute of Science an Technology, Nara, Japan. His research interests include speech and natural language processing, with a focus on machine learning approaches for applications such as machine translation, speech recognition, and spoken dialog.

Koichiro Yoshino received his B.A. degree in 2009 from Keio University, M.S. degree in informatics in 2011, and Ph.D. degree in informatics in 2014 from Kyoto University, respectively. From 2014 to 2015, he was a Research Fellow (PD) of Japan Society for Promotion of Science. Currently, he is an assistant professor in the Graduate School of Information Science, Nara Institute of Science and Technology. His research interests include spoken language processing, especially spoken dialogue system, syntactic and semantic parsing, and language modeling. Dr. Koichiro Yoshino received the JSAI SIG-research award in 2013. He is a member of IEEE, ACL, IPSJ, and ANLP.

Satoshi Nakamura received his B.S. from Kyoto Institute of Technology in 1981 and Ph.D. from Kyoto University in 1992. He was a director of ATR Spoken Language Communication Research Laboratories in 2000-2008, and a vice president of ATR in 2007-2008. He was a director general of Keihanna Research Laboratories, National Institute of Information and Communications Technology, Japan in 2009-2010. He is currently a professor and a director of Augmented Human Communication laboratory, Graduate School of Information Science at Nara Institute of Science and Technology. He is interested in modeling and systems of spoken dialog system, speech-to-speech translation. He is one of the leaders of speech-to-speech translation research projects including C-STAR, IWSLT and A-STAR. He headed the world first network-based commercial speech-to-speech translation service for 3-G mobile phones in 2007 and VoiceTra project for iPhone in 2010. He received LREC Antonio Zampoli Award, the Commendation for Science and Technology by the Ministry of Science and Technology in Japan. He is an elected board member of ISCA, International Speech Communication Association, and an elected member of IEEE SPS, speech and language TC.