Detecting Context Dependent Messages in a Conversational Environment

Chaozhuo Li[†], **Yu Wu**[†], **Wei Wu**[‡], **Chen Xing**[◇], **Zhoujun Li**[†], **Ming Zhou**[‡]
[†]State Key Lab of Software Development Environment, Beihang University, Beijing, China
[‡] Microsoft Research, Beijing, China
[◇] Nankai University, Tianjin, China
{lichaozhuo,wuyu,lizj}@buaa.edu.cn {wuwei,v-chxing,mingzhou}@microsoft.com

Abstract

While automatic response generation for building chatbot systems has drawn a lot of attention recently, there is limited understanding on when we need to consider the linguistic context of an input text in the generation process. The task is challenging, as messages in a conversational environment are short and informal, and evidence that can indicate a message is context dependent is scarce. After a study of social conversation data crawled from the web, we observed that some characteristics estimated from the responses of messages are discriminative for identifying context dependent messages. With the characteristics as weak supervision, we propose using a Long Short Term Memory (LSTM) network to learn a classifier. Our method carries out text representation and classifier learning in a unified framework. Experimental results show that the proposed method can significantly outperform baseline methods on accuracy of classification.

1 Introduction

Together with the rapid growth of social media such as Twitter and Weibo, the amount of conversation data on the web has tremendously increased. This makes building open domain chatbot systems with data-driven approaches possible. To carry on reasonable conversations with humans, a chatbot system needs to generate proper response with regard to users' messages. Recently, with the large amount of conversation data available, learning a response generator from data has drawn a lot of attention (Ritter et al., 2011; Shang et al., 2015; Vinyals and Le, 2015).

A key step to coherent response generation is determining when to consider linguistic context of messages. Existing work on response generation, however, has overlooked this step. They either totally ignores linguistic context (Ritter et al., 2011; Shang et al., 2015; Vinyals and Le, 2015) or simply considers context for every message (Sordoni et al., 2015b; Serban et al., 2015). The former case is easy to lead to irrelevant responses when users' input messages rely on the context information in previous conversation turns, while the latter case is costly (e.g., on memory and responding time) for building a real chatbot system and has the risk of bringing in noise to response generation especially when users want to end the current conversation topic and start a new one. According to our observation, there are two types of messages in a conversational environment. The first type is context dependent message, which means to reply to the message, one must consider previous utterances in the dialogue[1], while the second type is context independent message, which means even without the previous utterances, the message itself can still lead to a reasonable response. Table 1 compares the two types of messages using examples. In Case 1, "why do you think so" is a context dependent message. In order to reply to the message, one cannot ignore its linguistic context "I think it will rain tomorrow". On the other hand, in Case 2, "Well, what time is it now" is a context independent message, as one can give a reasonable response without looking at the previous turns. Distinguishing context dependent messages from context independent messages is important for building a good response generator. Missing linguistic context for context dependent

[1]Broadly speaking, context may not be limited to linguistic context. For example, a user's interest could also be a kind of context. As the first step, in this work, we only focus on "linguistic context".

Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 1990–1999, Osaka, Japan, December 11-17 2016.

Table 1: Two types of messages

Case 1 : a context dependent message	Case 2 : a context independent message
User : What will the weather be like tomorrow?	User : What are you doing?
Chatbot : I think it will rain tomorrow.	Chatbot : I am waiting for you to watch NBA.
User : ***Why do you think so?***	User : ***Well, what time is it now?***

messages will lead to nonsense response. For example, "because I love you" could also be a response for the message "why do you think so" if we only look at the message itself, but it is nonsense appearing in the dialogue of Case 1. Incorporating context information into context independent messages will increase the workload of a generation system and has the risk of bringing in noise to the generation process. For example, if we consider the context "NBA" for the message "Well, what time is it now", the chatbot will probably say something about "NBA" rather than answer the question with a time answer. Although detecting context dependent messages is crucial for building chatbot systems, there is limited understanding about it.

In this paper, we study this important but less explored problem. Instead of answering how to incorporate context information, we try to understand when we need the information. Therefore, our effort is complementary to the existing work on response generation. It can keep the existing generation algorithms context-aware and improve their efficiency and robustness to noise. The task is challenging, as messages in a conversational environment are usually short and informal, and evidence that can indicate a message is context dependent is scarce. For example, on 3 million post-response pairs crawled from Weibo, the average length of messages is 4.65. On such short texts, classic NLP tools such as POS Tagger and Parser suffer from bad performance (Derczynski et al., 2013; Foster et al., 2011) and it is difficult to explicitly extract features that are discriminative on the two types of messages. More seriously, there are no large scale annotations available for building a supervised learning procedure.

We consider leveraging the large amount of human-human conversation data available on the web to learn a message classifier. Our intuition is that a context dependent message has different linguistic context in different conversation sessions, therefore its responses could be more diverse on content than responses of a context independent message. To verify this idea, we study the distributions of responses of messages using conversation data crawled from social media and find that the length distribution of responses and the word distribution of responses are quite discriminative on the two types of messages. Based on this observation, for each message in the crawled data, we estimate the average length of responses, the entropy of the word distribution of responses, and the maximum mass of the word distribution of responses, and take these characteristics as weak supervision signals to learn a classifier. The classifier takes a message as input and can make prediction for any messages in a real conversation environment, even though the messages do not appear in the crawled data and characteristics like entropy are not available for them. We propose using a Long Short Term Memory (LSTM) architecture to learn the classifier. Our model represents message texts in a continuous vector space using a one-layer LSTM network. The text vectors are then provided as input to a two-layer feed-forward neural network to perform classification. The neural network architecture carries out feature learning and model learning in a unified framework, and thus can avoid explicit feature extraction which is difficult on short conversational messages. Our method leverages large scale weak supervision signals extracted from responses in social conversation data and can reach a satisfactory accuracy with only a few human annotations.

We conduct experiments on large scale English and Chinese conversation data mined from Twitter and Weibo respectively, and test the performance of our method on thousands of messages annotated by human labelers. Experimental results show that our method can significantly outperform baseline methods on accuracy of message classification on both of the two data sets.

We make the following contributions in this paper: 1) proposal of detecting context dependent messages in a conversational environment; 2) proposal of learning weak supervision signals from responses of messages using large scale conversation data; 3) proposal of using an LSTM architecture to learn a message classifier; 4) empirical verification of the proposed method on human annotated data.

2 Related Work

Our work lies in the path of building chatbot systems with data-driven approaches. Differing from traditional dialogue systems (cf., (Young et al., 2013)) which rely on hand-crafted features and rules to generate reply sentences for specific applications such as voice dialling (Williams, 2008) and appointment scheduling (Janarthanam et al., 2011) etc., recent effort focuses on exploiting an end-to-end approach to learn a response generator from social conversation data for open domain dialogue (Koshinda et al., 2015; Higashinaka et al., 2016). For example, Ritter et al. (Ritter et al., 2011) employed a phrase-based machine translation model for response generation. In (Shang et al., 2015; Vinyals and Le, 2015), neural network architectures were proposed to learning response generators from one-round conversation data. Based on these work, Sordoni et al. (Sordoni et al., 2015b) incorporated linguistic context into the learning of response generator. Serban et al. (Serban et al., 2015) proposed a hierarchical neural network architecture to building context-aware response generation. In this paper, instead of studying how to incorporate context into response generation, we consider the problem that when we need context in the process. Our work can keep the existing generation algorithms context-aware and at the same time improve their efficiency and robustness.

We employ a Recurrent Neural Network (RNN) architecture to learn a message classifier. RNN models (Elman, 1990), due to their capability of modeling sequences with arbitrary length, have been widely used in many natural language processing tasks such as language modeling (Mikolov et al., 2010) and tagging (Xu et al., 2015) etc. Recently, it is reported that Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU) (Cho et al., 2014) as two special RNN models which can capture long term dependencies in sequences outperform state of the art methods on tasks like machine translation (Sutskever et al., 2014) and response generation (Shang et al., 2015). In this paper, we apply the LSTM architecture to the task of context dependent message detection. We append LSTM with a two-layer feed-forward neural network, thus feature learning and model learning can be carried out simultaneously.

Our work belongs to the scope of short text classification (Song et al., 2014). Existing applications of short text classification include query classification (Kang and Kim, 2003), tweet classification (Sriram et al., 2010), and question classification (Zhang and Lee, 2003). We study a new problem in short text classification: distinguishing context dependent messages from context independent messages in a conversational environment. The task is important for building open domain chatbot systems and has its unique challenges (e.g., new data structure). We tackle the challenges by leveraging the responses of messages and utilizing an LSTM network to conduct feature learning and model learning simultaneously.

3 Learning to Detect Context Dependent Messages

Suppose that we have a data set $\mathcal{D} = \{(m_i, y_i)\}_{i=1}^{N}$ where m_i is a message composed of a sequence of words $(w_{m_i,1}, \ldots, w_{m_i,n_i})$ and y_i is an indicator whose value reflects whether m_i is context dependent or not. Our goal is to learn a function $g(\cdot) \in \{-1, 1\}$ using $\mathcal{D}$, thus for any new message m, $g(\cdot)$ predicts m a context dependent message if $g(m) = 1$. To this end, we need to answer two questions: 1) how to construct $\mathcal{D}$; 2) how to perform learning using $\mathcal{D}$.

For the first question, we can crawl conversation data from social media like Twitter and ask human labelers to annotate the messages in the data. The problem is that human annotation is expensive and time consuming and therefore we cannot obtain a large scale data set for learning. To solve the problem, we automatically learn some weak supervision signals using responses of messages in social conversation data, and take the signals as $\{y_i\}$ in $\mathcal{D}$. For the second question, one straightforward way is first extracting shallow features such as bag-of-words and syntax from messages and then employing off-the-shelf machine learning tools to learn a model. The problem is that shallow features are not effective enough on representing semantics in short conversation messages, which will be seen in our experiments. We propose using a Long Short Term Memory (LSTM) architecture to learn a model from $\mathcal{D}$. The advantage of our approach is that it can avoid explicit feature extraction and large scale human annotations, and carry out feature learning and model learning in a unified framework.

3.1 Learning Weak Supervision Using Responses

Instead of requiring human annotations, we consider creating signals that are discriminative on the two types of messages from large scale social conversation data available on the web. Our intuition is that a context dependent message has different linguistic context in different conversation sessions, therefore, its responses could be more diverse on content than responses of a context independent message (one message may appear multiple times, and therefore it may correspond to multiple responses). Table 2 illustrates our idea with some examples from Twitter. The last column of the table represents the frequency of the message or the frequency of the response under the message. For each message, we show the top 5 most frequent responses. From the examples, we can see that a context dependent message tends to have divergent and uniformly distributed responses corresponding to different linguistic context, while the responses of a context independent message share relatively similar content and some content dominates the distribution.

Table 2: Responses of the two types of messages

Context dependent message : why	2196	Context independent message : Good night	644
Response 1 : I am kidding	7	Response 1 : Good night	47
Response 2 : He can be like mcdaniels for sixer	5	Response 2 : Goodnight	44
Response 3 : Because I say no	5	Response 3 : Night	23
Response 4 : I am tired	5	Response 4 : Sleep well	10
Response 5 : U will become dependent on them	5	Response 5 : Thank you	9

The examples inspire us to investigate some statistical characteristics that can reflect the diversity of responses. These characteristics could be good indicators of context dependent messages, and we can construct $\{y_i\}$ in $\mathcal{D}$ using the characteristics. We estimate the following statistical characteristics for each message using its responses, and examine how the characteristics are discriminative on the two types of messages using 1000 labeled messages from Twitter and Weibo respectively. The details of the labeled data will be described in our experiments.

Entropy: the first characteristic we investigate is the entropy of the word distribution of responses, which is a common measure for diversity. Given a word distribution $P = (p_1, p_2, \ldots, p_n)$, the entropy of the distribution is defined as

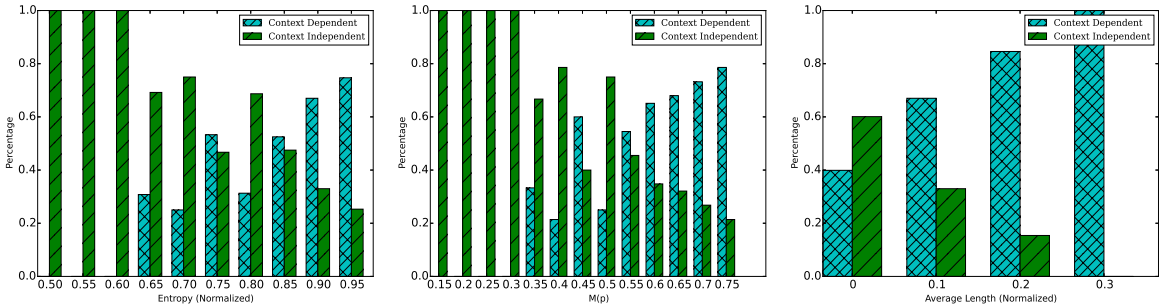
$$E(P) = \sum_{i=1}^{n} -p_i \log_2(p_i). \tag{1}$$

The maximum of the entropy is $\log_2(n)$ which is reached when the distribution is uniform. Then, a large entropy means a word distribution covers many words (i.e., n is big) and is close to a uniform distribution. Therefore, a context dependent message should have a larger entropy on responses than a context independent message (see the comparison in Table 2). We normalize the entropy to $[0, 1]$ by $\frac{E(P)-\min(E)}{\max(E)-\min(E)}$, where $\max(E)$ and $\min(E)$ represent the maximum entropy and the minimum entropy in the data set. Figure 1(a) shows the comparison of the two types of messages on normalized entropy using the Twitter labeled data. In the figure, each value on the x-axis represents an interval with a fixed length 0.05. For example, 0.50 means an interval $[0.5, 0.55)$. Each value on the y-axis represents the percentage of messages in a specific interval. For example, among messages falling in the interval $[0.95, 1)$, nearly 80% are labeled as context dependent and only about 20% are labeled as context independent. From the figure, we can see that entropy is discriminative on the two types of messages: context dependent messages distributes on large entropy areas, while context independent messages tend to have smaller entropy.

M(P): in addition to entropy, another characteristic that might reflect the diversity of responses could be the maximum mass of the word distribution of responses, as in diverse responses, words should be uniformly distributed (stopwords are removed), while in less diverse responses, there may exit dominant words (e.g., "night" in Table 2). Given a word distribution $P = (p_1, p_2, \ldots, p_n)$, we define a characteristic as

$$M(P) = 1 - \max_{1 \leqslant i \leqslant n} p_i \tag{2}$$

Figure 1(b) compares the two types of messages on $M(P)$ using the Twitter labeled data, in which values



(a) Comparison on entropy

(b) Comparison on $M(P)$

(c) Comparison on average length of responses

Figure 1: Comparison of the two types of messages on three characteristics.

on the x-axis and y-axis have the same meaning as those in Figure 1(a). From the figure, we can see that similar to entropy, $M(P)$ is useful on distinguishing the two types of messages. Context dependent messages have larger $M(P)$ than context independent messages.

Average length of responses: finally, we consider the length distribution of responses. Since responses of context dependent messages are more diverse on content, they might be longer than responses of context independent messages. We calculate the average length of responses for each message and normalize it to $[0, 1]$ in the same way as entropy. Figure 1(c) compares the two types of messages on average length of responses using the Twitter labeled data, where values on the x-axis represent intervals with a length 0.1. The result supports our claim and clearly indicates that average length is discriminative on the two types of messages.

We combine the three characteristics using a linear SVM classifier learned with the 1000 labeled messages and take the output of the SVM (a real value) as $\{y_i\}$ in $\mathcal{D}$. By this means, we can create a large scale training data set with only a little human labeling effort. Here, as a reference, we also report the classification accuracy of the three characteristics and the SVM classifier on the 1000 labeled data. Each characteristic corresponds to a threshold tuned on the 1000 labeled data with 5-fold cross validation. If a value of a characteristic of a message is larger than the threshold, then the message will be predicted as context dependent. Table 3 shows the classification accuracy of 5-fold cross validation (average of 5 results), where SVM (com) refers to the SVM classifier. Details of experiment setting will be described in Section 4. From Table 3, we can see that the numbers are consistent with Figure 1(a), 1(b), and 1(c).

Table 3: Classification accuracy on 1000 labeled data

	Weibo	Twitter
Entropy	72.6 %	70.5 %
$M(P)$	72.6 %	69.8 %
Average length of responses	72.8 %	68.5 %
SVM (com)	73.8 %	71.2 %

3.2 Model Learning

We head for learning $g(\cdot)$ using $\mathcal{D}$ constructed in Section 3.1. Note that $g(\cdot)$ only takes a message m as input, and thus can make prediction for any messages in a real chatbot system even though the messages are not in $\mathcal{D}$ and their entropy, M(P), and average length of responses are not available. Our idea is that we first learn a regression model by fitting $\{y_i\}$ in $\mathcal{D}$ through minimizing the sum of squared residuals and then construct $g(\cdot)$ by comparing the output of the regression model with a threshold. We can obtain the threshold by tuning it on a few labeled data (e.g., the 1000 labeled data). The key is how to learn the regression model. We propose using a Recurrent Neural Network (RNN) architecture to embed messages into a continuous vector space and learning a regression model with the embedding of messages using a feed-forward neural network. The RNN model, which is capable of embedding sequences with arbitrary

length, can encode the order of words and the semantics of a message into a vector representation which has been recently proven effective on capturing similarity of short texts (Sordoni et al., 2015a). We take the output vector given by RNN as a feature representation of a message and feed it to a feed-forward neural work. By this means, we can conduct feature learning and model learning in a unified framework and jointly optimize the two components.

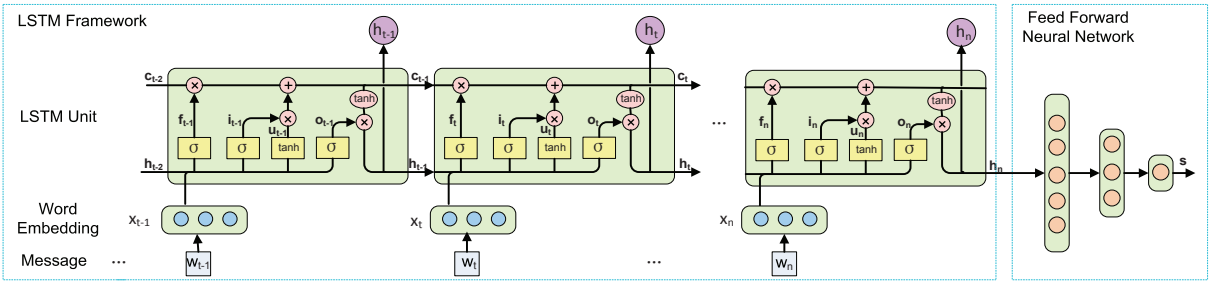


Figure 2: The architecture of our method

Given a message m which consists of n words, the RNN model reads the words one by one, and updates a recurrent state h_t for the t-th word w_t by

$$h_t = f(h_{t-1}, x_t),\ h_0 = 0, \tag{3}$$

where $h_t \in \mathbb{R}^{d_h}$, $x_t \in \mathbb{R}^{d_w}$ is the vector representation of w_t, and f is non-linear transformation. h_t acts as an encoding of the semantics of the word sequence up to position t, and the final output h_n is a representation of message m. Both x_t and h_t are learned in the optimization of the RNN model. We select the Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) as f, since it can model long term dependencies in sequences with affordable complexity. LSTM controls the learning of the representation of a sequence by gates. Specifically, at position t, LSTM controls the information that should be kept from previous states by an input gate i_t, and the information that should be forgotten by a forget gate f_t. After memorizing and forgetting, the information is stored in a memory cell c_t. c_t generates the recurrent state h_t through an output gate o_t. The specific parameterization of LSTM is given by

$$\begin{aligned}
i_t &= \sigma(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)}) \\
f_t &= \sigma(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}) \\
o_t &= \sigma(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}) \\
u_t &= \tanh(W^{(u)}x_t + U^{(u)}h_{t-1} + b^{(u)}) \\
c_t &= i_t \otimes u_t + f_t \otimes c_{(t-1)} \\
h_t &= o_t \otimes \tanh(c_t),
\end{aligned}$$

where $\sigma(\cdot)$ is a sigmoid function and $\tanh(\cdot)$ is a hyperbolic tangent function. $W^{(i)}$, $W^{(f)}$, $W^{(o)}$, $W^{(u)}$ $\in \mathbb{R}^{d_h \times d_w}$, $U^{(i)}$, $U^{(f)}$, $U^{(o)}$, $U^{(u)} \in \mathbb{R}^{d_h \times d_h}$, and $b^{(i)}$, $b^{(f)}$, $b^{(o)}$, $b^{(u)} \in \mathbb{R}^{d_h \times 1}$ are parameters. $\otimes$ means element-wise multiplication. After we get the final state h_n, we feed it to a two-layer feed-forward neural network to get an output s which is defined by

$$s = b_2 + W_2\left(\tanh(b_1 + W_1 h_n)\right), \tag{4}$$

where $b_1 \in \mathbb{R}^{d_s \times 1}$, $W_1 \in \mathbb{R}^{d_s \times d_h}$, $W_2 \in \mathbb{R}^{1 \times d_s}$, and $b_2 \in \mathbb{R}$ are parameters. Figure 2 illustrates the architecture of our method.

For each m_i in $\mathcal{D}$, we calculate an s_i using Equation (4) as an estimation of y_i. We then learn the parameters of the LSTM network and the feed-forward network by minimizing the sum of the squared residuals. Formally, our learning approach can be formulated as

$$\arg\min_{s} \sum_{i=1}^{N} (y_i - s_i)^2. \tag{5}$$

After we obtain the parameters, we can calculate an s_m for any message m using Equation (4). We then tune a threshold T with a few labeled messages. The classifier $g(\cdot)$ is given by

$$g(m) = \begin{cases} 1 & \text{if } s_m > T \\ -1 & \text{otherwise} \end{cases} \quad (6)$$

The gradients of the objective function (5) are computed using the back-propagation through time (BPTT) algorithm (Williams and Peng, 1990). We share the code for model learning at `https://github.com/whatsname1991/coling2016`.

4 Experiments

4.1 Experiment Setup

We constructed the conversation data for experiments from Weibo and Twitter. In each of the two social media, two persons can communicate by replying to each other under a post. We crawled sequences of reply with posts and extracted triples like "(context, message, response)" as experimental data. In a triple, "message" is a reply, "context" is the sentence in the previous turn of the message (a reply or a post), and "response" is the sentence in the next turn (reply to the message). Note that in this work, we restrict the context of a message to a single sentence. This is a simplification of context in conversation. In real conversation, context could be more complicated and we leave the discussion of it as future work.

We crawled 5.9 million English triples from Twitter, and 3.1 million Chinese triples from Weibo. The numbers of distinct messages in the Twitter data and in the Weibo data are $92,755$ and $112,175$ respectively. On average, each Twitter message has 63.26 responses (some messages like "hello" can have many different responses) and each Weibo message has 27.52 responses. The average word length of Twitter message is 3.39 and the word average length of Weibo message is 4.65. English sentences were stemmed and stop words were removed, and Chinese sentences were segmented.

We constructed $\mathcal{D} = \{(m_i, y_i)\}_{i=1}^N$ in Section 3.1 in the following way: we first calculated entropy, $M(P)$, and average length of responses for each message using the 5.9 million English triples and 3.1 million Chinese triples. Then from these data, we randomly sampled 1000 English triples and 1000 Chinese triples as validation sets. For each triple in the validation data, we hid the response and recruited human judges to label if the message is context dependent or not. Note that we hid responses when labeling messages because this is more close to the real case. In a real chatbot system, one has to determine if a message is context dependent or not before generating a response. Each judge labeled a message with 1 if it is context dependent, otherwise the judge labeled the message with -1. Each message got three labels and the majority of the labels was taken as the final decision for the message. In the Weibo data, there are 412 positive examples and 588 negative examples. In the Twitter data, the two numbers are 440 and 560, respectively. With the two validation data sets, we learned two SVM classifiers in order to combine the three characteristics as described in Section 3.1. Parameters of SVMs were tuned by 5-fold cross validation. Finally, we assigned a y_i to each m_i in the $112,175$ Twitter messages and $92,755$ Weibo messages by the output of the SVM classifiers, and formed $\mathcal{D}$ for both English data and Chinese data. We trained LSTM models using $\mathcal{D}$.

To evaluate the performance of different models, we crawled another 3000 Chinese context-message pairs and 1000 English context-message pairs from Weibo and Twitter respectively, and followed the same way as the validation data to judge if the messages are context dependent or not. We used these data to simulate real context-message pairs in chatbot systems. In the Weibo data, there are 2715 unique messages and 1983 messages are not in $\mathcal{D}$. The numbers of positive examples and negative examples are 1472 and 1528 respectively. In the Twitter data, the number of unique messages is 875 and 366 messages are not included by $\mathcal{D}$. The numbers of positive and negative examples are 464 and 536 respectively. Note that for messages that are not included by $\mathcal{D}$, their characteristics (i.e., entropy, $M(P)$, and average length of responses) are not available, and we can only use classifiers whose features are extracted from messages (like our LSTM models) to make prediction. This is close to a real situation in chatbots, and we took the two data sets as test sets.

We considered the following methods as baselines:

Length: intuitively, short messages tend to be context dependent (e.g., "why" in Table 2). Therefore, we employed length of a message as a baseline. A message shorter than a threshold will be predicted as a context dependent message.

MDF: given a word, we estimated the number of messages that contain the word and named it "document frequency" (DF). We constructed a list of words associated with DF using $\mathcal{D}$. For a new message, we calculated the minimal DF of words in the message using the list. A context dependent message like "why do you think so" may consist of common words, and thus correspond to a high minimal DF. We considered minimal DF as a baseline. A message with a minimal DF larger than a threshold will be predicted as a context dependent message.

SVM (Length+MDF): we linearly combined Length and MDF by learning an SVM classifier on the validation data.

SVM (classification): we extracted unigrams, bigrams, and frequencies of POS tags as features from a message, and learned a linear SVM classifier on the validation data with these features. POS tags for Chinese data were obtained using Stanford Parser (`http://nlp.stanford.edu/software/lex-parser.shtml`) and POS tags for English data were obtained using TweetNLP (`http://www.cs.cmu.edu/~ark/TweetNLP/`).

SVM (regression): instead of learning a classifier from annotations in the validation data, we fitted $\{y_i\}$ in $\mathcal{D}$ by learning an SVM regression model using the same features as SVM (classification) and made predictions on new messages by a threshold.

All SVM models were learned using SVM-Light (`http://svmlight.joachims.org/`). We employed classification accuracy as an evaluation metric.

4.2 Parameter Tuning

For Length and MDF, the only parameter is a threshold. We tuned the thresholds on the validation data. For all SVM models, we selected the trade-off parameter in SVM from $\{0.01, 0.1, 1, 10, 100\}$ by 5-fold cross validation on the validation data. SVM (regression) also needs a threshold. We tuned it on the validation data. The parameters of LSTM include the dimension of word vectors d_w, the dimension of hidden states d_h, and the dimension of the first layer of the feed-forward network d_s. We set $d_w = d_h = 256$, and $d_s = 100$. Besides these parameters, we also set a dropout rate 0.1 in the learning of the feed-forward network as regularization.

Table 4: Accuracy on two test sets

	Weibo	Twitter
Length	62.6 %	61.3 %
MDF	62.1 %	58.6 %
SVM (Length+MDF)	63.0 %	62.2 %
SVM (classification)	66.8 %	65.4 %
SVM (regression)	64.3 %	68.3 %
LSTM	**75.6 %**	**73.4 %**

Table 5: Comparison between LSTM, SVM (classification), and SVM (regression)

Example	context : Have you heard Taylor Swift's new song? *message: Yep, I have heard it on Saturday night.*
Label	context dependent
SVM (regression)	context independent
SVM (classification)	context independent
LSTM	context dependent

4.3 Quantitative Evaluation

Table 4 reports quantitative evaluation results on the test data. From the results, we can see that our methods outperform baseline methods. The improvement over the best performing baseline methods (i.e., SVM (classification) on Webio and SVM(regression) on Twitter) is statistically significant (sign test, p-value < 0.01).

Length and MDF are characteristics of messages. The results tell us that these characteristics are not so discriminative on the two types of messages. The reason is easy to understand: we may think that context dependent messages tend to be short and consist of common words, but the fact is that short messages composed of common words could be context independent (e.g., "Good night" in Table 2) while long messages like "Yep, I have heard it on Saturday night" (see the example in Table 5) could be context

dependent. Both SVM (classification) and SVM (regression) perform worse than our LSTM model, indicating that shallow features are not effective enough to represent the semantics in short conversation messages. Our method outperforms the baseline methods on both data sets. The results verified our idea on leveraging responses for context dependent message detection, and demonstrates the power of big data and the advantage of LSTM on capturing semantics in short messages.

4.4 Qualitative Evaluation

We use an example to further explain why our method is effective on distinguishing the two types of messages. Table 5 compares LSTM with SVM (classification) and SVM (regression). Both SVM (classification) and SVM (regression) rely on shallow features such as bag of words and pos tags to perform learning. These features, however, are not effective on representing the semantics of short messages. The representation is easily to be biased by some specific words like "Saturday night" in the example. Therefore, both SVM (classification) and SVM (regression) failed on this case. On the other hand, LSTM models term dependencies in sequences with a memorizing-forgetting mechanism. It can capture the semantics in the message "Yep, I have heard it on Saturday night." and identify that it is similar to messages like "Yes, I did" and "Yes, I have". For example, the cosine of the vector of "Yep, I have heard it on Saturday night." and the vector of "Yes, I have" given by LSTM is 0.63. Since messages like "Yes, I did" and "Yes, I have" are common context dependent messages, LSTM can successfully recognize that the message in the example is also context dependent.

5 Conclusion

We propose learning a LSTM network with weak supervision signals estimated from responses of messages to detecting context dependent messages in a conversational environment. Evaluation results show that the proposed method can significantly outperform baseline methods on distinguishing the two types of messages.

Acknowledgement

This work was supported by Beijing Advanced Innovation Center for Imaging Technology (No.BAICIT-2016001), the National Natural Science Foundation of China (Grand Nos. 61370126, 61672081), National High Technology Research and Development Program of China (No.2015AA016004),the Fund of the State Key Laboratory of Software Development Environment (No.SKLSDE-2015ZX-16).

References

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *EMNLP*.

Leon Derczynski, Alan Ritter, Sam Clark, and Kalina Bontcheva. 2013. Twitter part-of-speech tagging for all: Overcoming sparse and noisy data. In *RANLP*, pages 198–206.

Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science*, 14(2):179–211.

Jennifer Foster, Özlem Çetinoglu, Joachim Wagner, Joseph Le Roux, Stephen Hogan, Joakim Nivre, Deirdre Hogan, and Josef Van Genabith. 2011. # hardtoparse: Pos tagging and parsing the twitterverse. In *AAAI 2011 Workshop on Analyzing Microtext*, pages 20–25.

Ryuichiro Higashinaka, Nozomi Kobayashi, Toru Hirano, Chiaki Miyazaki, Toyomi Meguro, Toshiro Makino, and Yoshihiro Matsuo. 2016. Syntactic filtering and content-based retrieval of twitter sentences for the generation of system utterances in dialogue systems. In *Situated Dialog in Speech-Based Human-Computer Interaction*, pages 15–26. Springer.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Srinivasan Janarthanam, Helen Hastie, Oliver Lemon, and Xingkun Liu. 2011. The day after the day after tomorrow?: a machine learning approach to adaptive temporal expression generation: training and evaluation with real users. In *Proceedings of the SIGDIAL 2011 Conference*, pages 142–151. Association for Computational Linguistics.

In-Ho Kang and GilChang Kim. 2003. Query type classification for web document retrieval. In *SIGIR*, pages 64–71. ACM.

Makoto Koshinda, Michimasa Inaba, and Kenichi Takahashi. 2015. Machine-learned ranking based non-task-oriented dialogue agent using twitter data. In *2015 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, volume 3, pages 5–8. IEEE.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 1045–1048.

Alan Ritter, Colin Cherry, and William B Dolan. 2011. Data-driven response generation in social media. In *EMNLP*, pages 583–593. Association for Computational Linguistics.

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2015. Building end-to-end dialogue systems using generative hierarchical neural network models. *arXiv preprint arXiv:1507.04808*.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. *arXiv preprint arXiv:1503.02364*.

Ge Song, Yunming Ye, Xiaolin Du, Xiaohui Huang, and Shifu Bie. 2014. Short text classification: A survey. *Journal of Multimedia*, 9(5):635–643.

Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015a. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 553–562. ACM.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015b. A neural network approach to context-sensitive generation of conversational responses. *arXiv preprint arXiv:1506.06714*.

Bharath Sriram, Dave Fuhry, Engin Demir, Hakan Ferhatosmanoglu, and Murat Demirbas. 2010. Short text classification in twitter to improve information filtering. In *SIGIR*, pages 841–842. ACM.

Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Ronald J Williams and Jing Peng. 1990. An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2(4):490–501.

Jason Williams. 2008. Demonstration of a pomdp voice dialer. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Demo Session*, pages 1–4. Association for Computational Linguistics.

Wenduan Xu, Michael Auli, and Stephen Clark. 2015. Ccg supertagging with a recurrent neural network. In *ACL'15*, volume 2, pages 250–255.

Stephanie Young, Milica Gasic, Blaise Thomson, and John D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179.

Dell Zhang and Wee Sun Lee. 2003. Question classification using support vector machines. In *SIGIR*, pages 26–32. ACM.