Elastic Responding Machine for Dialog Generation with Dynamically Mechanism Selecting

Conference Paper · February 2018

CITATION
1

READS
270

6 authors, including:

Ganbin Zhou
Chinese Academy of Sciences
4 PUBLICATIONS **5** CITATIONS
SEE PROFILE

Qing He
Chinese Academy of Sciences
157 PUBLICATIONS **1,520** CITATIONS
SEE PROFILE

Elastic Responding Machine for Dialog Generation with Dynamically Mechanism Selecting

Ganbin Zhou[1,2], **Ping Luo**[1,2], **Yijun Xiao**[3], **Fen Lin**[4], **Bo Chen**[4], **Qing He**[1,2]
[1]Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS),
Institute of Computing Technology, CAS, Beijing 100190, China.{zhouganbin, luop, heqing}@ict.ac.cn
[2]University of Chinese Academy of Sciences, Beijing 100049, China.
[3]Department of Computer Science, University of California Santa Barbara, Santa Barbara, CA 93106, USA.
[4]WeChat Search Application Department, Tencent, China.

Abstract

Neural models aiming at generating meaningful and diverse response is attracting increasing attention over recent years. For a given post, the conventional encoder-decoder models tend to learn high-frequency but trivial responses, or are difficult to determine which speaking styles are suitable to generate responses. To address this issue, we propose the elastic responding machine (ERM), which is based on a proposed *encoder-diverter-filter-decoder* framework. ERM models the multiple responding mechanisms to not only generate acceptable responses for a given post but also improve the diversity of responses. Here, the mechanisms could be regraded as some latent variables, and for a given post different responses may be generated by different mechanisms. The experiments demonstrate the quality and diversity of the generated responses, intuitively show how the learned model controls response mechanism when responding, and reveal some underlying relationship between mechanism and language style.

Introduction

Recent years have witnessed the development of conversational models which aim at the generation of relevant and fluent responses observing user posts. Since the end-to-end neural machine automatically learns the transduction from posts to responses, it is attracting increasing number of studies for developing task-oriented as well as open-domain conversation systems. The vast amount of dialogue texts generated by social networks provide the data basis for generative models of conversational models, and these end-to-end solutions are shown to outperform the conventional ones (Shang, Lu, and Li 2015).

Despite the surging popularity of these end-to-end neural machines, their performances are still far from perfect. In real human to human conversations, different language styles exist. For example, some people compose their speech with plain declarative statements; some prefer to use negations; others fill their responses with rhetorical questions. Unlike real conversations, end-to-end neural models often generate generic and dull responses, and fail in controlling the language styles of responses. Current generative conversational models follows the line of encoder-decoder framework for statistic machine translation (SMT). These models attempt to "translate" an input post $\boldsymbol{x}$ to a response $\boldsymbol{y}$ by maximizing the probability of $p(\boldsymbol{y}|\boldsymbol{x})$. A training objective of maximum likelihood tends to produce low-diversified high-frequency responses (Li et al. 2016a). Although beam search algorithm applied in these models offers capability of generating multiple probable responses to a given input, there is no guarantee that these responses are diversified in terms of language style.

To improve response diversity, models have been proposed including making improvements on the beam search method (Wiseman and Rush 2016), designing new types of objective functions for response resorting (Li et al. 2016a), etc. Specially, Zhou et al. (2017) argues that a machine translation corpus is inherently different from a conversation corpus. For translation pairs, there almost always is a phrase alignment between each pair and there is little variation among the target translations for a given source sentence. On the other hand, for conversation corpus, aside from the language style elements mentioned earlier, responses can be wildly different from one another focusing on various topics in the input post. Therefore, they proposed a encoder-diverter-decoder framework MARM which models these variations as latent mechanism parameters. Here, mechanism refers to the different underlying transduction regularities from a post to its corresponding response. With different learned mechanisms specified, the model is able to generate responses with various language styles.

In (Zhou et al. 2017), it is assumed that there are a fixed number of mechanisms for all pairs of posts and responses. During training, for each input post the model always calculates the conditional probability $p(\boldsymbol{y}|m, \boldsymbol{x})$ over all mechanisms. However, it is possible that not all mechanisms are relevant for a given input post $\boldsymbol{x}$. This introduces an inefficiency in computing: the over-all-mechanism training applies back-propagation for all mechanism parameters including those that are not related to $\boldsymbol{x}$. Thus, there is more room for improvements with respect to time efficiency.

At test time, MARM utilizes top-K mechanisms for each input post. However, we argue that it is difficult and unreliable to manually select the hyper-parameter K, since the number of suitable mechanisms varies for different input posts. For example, for posts such as "I have an exam tomorrow and I am super nervous", the responses could be "What exam?", "Relax, it is quite easy.", or "You should not

be. That exam is not hard.". On the other hand, certain posts only have specific responses, such as "Hello!", "It was nice meeting you.". Their generating mechanisms are less than the aforementioned ones. It is vital to be able to dynamically determine the number of mechanisms on the fly after observing the input post.

In this study, we plan to enhance mechanism-aware neural machine by adding the capability to dynamically select the number of mechanisms for each individual input post therefore improving computation efficiency and reducing mechanism redundancy. We propose elastic responding machine (ERM) which is able to automatically determine which mechanisms should be used in the responding phase for a specific input post. The key idea is to train a model to learn a large set of mechanisms and only select a suitable subset of mechanisms for responding a given post. Since each mechanism often obtains a specific type of language style or genre, this suggests a more elegant way to control which language styles will be used when responding. This could also effectively reduce the computational complexity for each batch and enable the model to provide the capability of training a larger set of mechanisms to potentially improve the performance over limited computing resource (e.g. GPU memory). As it is non-trivial to label the training corpus with mechanism information, we train the model using only the post-response pairs. To this end, our contribution is summarized into three folds:

1) We propose a encoder-diverter-filter-decoder framework, in which the suitable mechanism set is selected to generate diverse responses while the mechanism redundancy could be minimized. 2) We empirically demonstrate that the proposed method generates more diverse and acceptable responses than baseline methods. Specifically, since ERM may select suitable mechanisms from a large set of mechanisms, it achieves a 7.00% increase of acceptance ratio and 7.41% increase of diversity-F1. 3) We investigate the diversity distribution of the corpus, and explore the relationship between language style and mechanism.

Preliminaries

Encoder-Decoder

Given a post $\boldsymbol{x} = (x_1, x_2, \cdots, x_T)$ and a response $\boldsymbol{y} = (y_1, y_2, \cdots, y_{T'})$, where x_t and y_t are the t-th word in post and response respectively. The encoder-decoder model (Cho et al. 2014; Sutskever, Vinyals, and Le 2014) aims to learn $p(\boldsymbol{y}|\boldsymbol{x})$ based on the training corpus $D = \{(\boldsymbol{x}, \boldsymbol{y})\}$ containing post-response pairs.

In detail, the conventional encoder-decoder model firstly uses encoder module to summarize the post as a fixed-length vector representation, namely context vector $\boldsymbol{c}$. Then it feeds $\boldsymbol{c}$ to a decoder for generating responses. Here, both encoder and decoder are recurrent neural networks. For encoder, it receives the previous hidden state $\boldsymbol{h}_{t-1}$ and current word embedding $\boldsymbol{x}_t$, and calculate $\boldsymbol{h}_t = f(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t)$, where f is the activation function, e.g. LSTM (Hochreiter and Schmidhuber 1997), GRU (Cho et al. 2014) etc. At last, the encoder output the last hidden state $\boldsymbol{h}_T$ as $\boldsymbol{c}$, representing the summarization of the post. Similar to the encoder, the decoder also recurrently generate the hidden states $\boldsymbol{s}_t$. Additionally, the decoder feeds every s_t to a softmax layer to estimate the word probability $p(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$.

Encoder-Diverter-Decoder

In consideration of 1-to-n relationships between a post to its diverse responses, the encoder-diverter-decoder model (Zhou et al. 2017) is proposed to model different responding mechanisms. This model assumes that there are M latent mechanisms $\{m\}_{i=1}^{M}$ for response generation. Thus, the $p(\boldsymbol{y}|\boldsymbol{x})$ can be decomposed as

$$p(\boldsymbol{y}|\boldsymbol{x}) = \sum_{i=1}^{M} p(m_i|\boldsymbol{x}) p(\boldsymbol{y}|m_i, \boldsymbol{x}) \tag{1}$$

where $p(m_i|\boldsymbol{x})$ measures the degree that m_i can respond the post x, and $p(\boldsymbol{y}|m_i, \boldsymbol{x})$ measures the probability that the response $\boldsymbol{y}$ is generated by the mechanism m_i and post $\boldsymbol{x}$. In detail, as conventional encoder-decoder model, its encoder firstly summarizes the post as a fixed-length vector $\boldsymbol{c}$. Then, a diverter, which is a softmax classifier, receives $\boldsymbol{c}$ as input, and output the probability $p(m_i|\boldsymbol{x})$. Here, it is calculated as $p(m_i|\boldsymbol{x}) = \text{softmax}(\boldsymbol{m}_i \cdot \boldsymbol{c} + b_i)$ where $\boldsymbol{m}_i$ is mechanism embedding. $p(m_i|\boldsymbol{x})$ will be further used for mechanism selecting. To generate a response, for a given m_i and $\boldsymbol{x}$, the decoder receives a mechanism-aware context vector $\widetilde{\boldsymbol{c}}_i = [\boldsymbol{m}_i; \boldsymbol{c}]$ which is the concatenation of mechanism embedding $\boldsymbol{m}_i$ and context vector $\boldsymbol{c}$. For a given $\widetilde{\boldsymbol{c}}_i$, the decoder updates the hidden state $\boldsymbol{s}_t = f(\boldsymbol{s}_{t-1}, y_{t-1}, \widetilde{\boldsymbol{c}}_i)$, and uses s_t to estimate the word probability $p(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$.

In Equ.(1), $p(m|\boldsymbol{x})$ represents the probability of the mechanism m conditioned on $\boldsymbol{x}$. This probability actually measures the degree that m can generate the response for $\boldsymbol{x}$. The larger of this value is, the more likely that the mechanism m_i can be used to respond $\boldsymbol{x}$.

Elastic-mechanism Responding Machine

Following the line of mechanism-aware responding machine, we propose a new framework Elastic Responding Machine (ERM), which dynamically determines the mechanism used for responding.

The model assumes that in the corpus all mechanisms are contained in the set $\mathcal{S} = \{m_i\}_{i=1}^{M}$. Denote an additional termination mechanism m_0 (detailed later). For a given post $\boldsymbol{x}$, the model selects a suitable mechanism subset $\mathcal{S}_{\boldsymbol{x}} \in \mathcal{S}$ and utilizes $\mathcal{S}_{\boldsymbol{x}}$ to generate $\boldsymbol{x}$'s response, namely $\boldsymbol{y}$. The generative probability of $\boldsymbol{y}$ is modeled as

$$p\big(\boldsymbol{y}|\boldsymbol{x}\big) = \frac{\sum_{m \in \mathcal{S}_{\boldsymbol{x}}} p(m|\boldsymbol{x}) p(\boldsymbol{y}|m, \boldsymbol{x})}{\sum_{m \in \mathcal{S}_{\boldsymbol{x}}} p(m|\boldsymbol{x})} \tag{2}$$

where the denominator normalizes the probability sum of $p\big(m \in \mathcal{S}_{\boldsymbol{x}}|\boldsymbol{x}\big)$ to 1.

To model $p(\boldsymbol{y}|\boldsymbol{x})$, the encoder firstly summarizes the input post $\boldsymbol{x}$ as the context embedding $\boldsymbol{c}$. Due to space limitation, we omit the encoder details. Then the context embedding $\boldsymbol{c}$ is fed to the diverter to calculate $\boldsymbol{x}$'s distribution over all

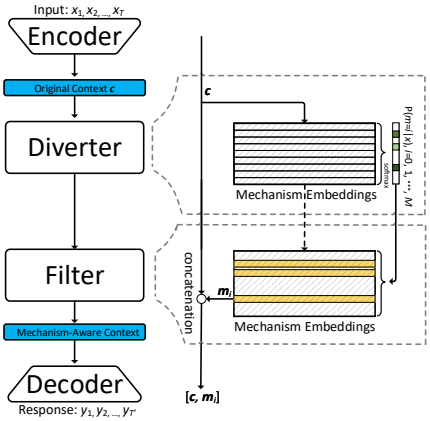


Figure 1: Structure of encoder-diverter-filter-decoder model.

mechanisms. Specially, for all mechanisms $p(m_i|\boldsymbol{x})$ can be modeled as follows,

$$p(m_i|\boldsymbol{x}) = \frac{\exp g(\boldsymbol{m}_i, \boldsymbol{c})}{\sum_{j=0}^{M} \exp g(\boldsymbol{m}_j, \boldsymbol{c})}, i = 0, 1, \cdots, M \quad (3)$$

where g can be any nonlinear, potentially multi-layered function and $\boldsymbol{m}_i$ represents the embedding of the i-th mechanism. $\{\boldsymbol{m}_i\}_{i=0}^{M}$ are trained as model parameters.

Mechanism Selecting

We now move to discuss how to select a suitable mechanism set $\mathcal{S}_{\boldsymbol{x}}$ to generate responses. With calculated $p(m_i|\boldsymbol{x})$, we then develop a filter component aiming at selecting which mechanisms can be used for response generation.

Here, a filter component is developed to receive the mechanism probability $p(m|\boldsymbol{x})$, and select suitable mechanisms for responding $\boldsymbol{x}$ in a sequential fashion(Fig.1), and we train the filter via reinforcement algorithm. Formally, with mechanism probability $p(m_i|\boldsymbol{x})$ the filter selects a mechanism set $\mathcal{S}_{\boldsymbol{x}} \in \mathcal{S}$. Note that an ideal $\mathcal{S}_{\boldsymbol{x}}$ should satisfy: 1) $\mathcal{S}_{\boldsymbol{x}}$ include enough mechanisms for $\boldsymbol{x}$. Hence, the diversity of generated responses will be high. 2) The mechanisms in $\mathcal{S}_{\boldsymbol{x}}$ are not highly overlapped or redundant. In other words, for each mechanism its generated response is different from other mechanisms.

Now we introduce how the filter selects $\mathcal{S}_{\boldsymbol{x}} \in \mathcal{S}$. The filter sorts the mechanisms in descending order of $p(m_i|\boldsymbol{x})$ (denoted as $m_{i_1}, m_{i_2}, \cdots, m_{i_M}$). Then the filter scans the mechanisms from m_{i_1} to m_{i_M}, and selects only top-K mechanisms ($m_{i_1}, m_{i_2}, \cdots, m_{i_K}$). In detail, the filter composed of the following components:

Action: We define two actions in our model: a termination action a_t and continue action a_c representing the filter will terminate or continue selecting.

State: The state $\mathcal{S}_{\boldsymbol{x}}^{(\cdot)}$ is the set of selected mechanisms. Here, let $\mathcal{S}_{\boldsymbol{x}}^{(k)} \in \mathcal{S}$ denote the state at k-th step. $\mathcal{S}_{\boldsymbol{x}}^{(k)}$ denotes the mechanism set with top-k probability, namely, $\mathcal{S}_{\boldsymbol{x}}^{(k)} = \{m_{i_j}\}_{j=1}^{k}$. Specially, the initial state $\mathcal{S}_{\boldsymbol{x}}^{(1)}$ is the set only containing m_{i_1} which obtains the highest probability $p(m_{i_1}|\boldsymbol{x})$.

Algorithm 1 FILTER$(\boldsymbol{x}, \mathcal{S})$

Input:
Post, $\boldsymbol{x}$
Total mechanism set, $\mathcal{S}$
Output: Selected mechanisms for input $\boldsymbol{x}$, $\mathcal{S}_{\boldsymbol{x}}$

```
Sort the mechanisms m ∈ S in the descending order of p(m_i|x) (denoted as m_{i_1}, m_{i_2}, ..., m_{i_M});
Set S_x^(0) ← φ;
for k ← 1 to |S| do
    S_x^(k) ← S_x^(k-1) + {m_{i_k}};
    Sample a ~ π(·|S_x^(k));
    if a = a_t then
        break;
    end if
end for
K ← k; S_x ← S_x^(K);
return S_x;
```

Policy: Given the current state, the filter determines whether the selecting process should be terminated. If terminated, the filter will output current selected mechanisms. Here, given the state $\mathcal{S}_{\boldsymbol{x}}^{(k)}$, the two actions ($a_c$ and a_t) are taken by a stochastic policy $\pi(\cdot|\mathcal{S}_{\boldsymbol{x}}^{(k)})$. To obtain $\pi(\cdot|\mathcal{S}_{\boldsymbol{x}}^{(k)})$, we firstly define the m_0 as the termination mechanism. Then we define:

$$\begin{aligned} \pi(a_t|\mathcal{S}_{\boldsymbol{x}}^{(k)}) &= \frac{p(m_0|\boldsymbol{x})}{1 - \sum_{m \in \mathcal{S}_{\boldsymbol{x}}^{(k)}} p(m|\boldsymbol{x})} \\ \pi(a_c|\mathcal{S}_{\boldsymbol{x}}^{(k)}) &= 1 - \pi(a_t|\mathcal{S}_{\boldsymbol{x}}^{(k)}) \end{aligned} \quad (4)$$

Here, $\pi(a_t|\mathcal{S}_{\boldsymbol{x}}^{(k)})$ is defined as the ratio of the probability of termination mechanism m_0 with respect to the sum probability of unselected mechanisms. Hence, the filter tends to continue selecting if: 1) the number of unselected mechanism is still "large"; 2) there are unselected mechanisms with high probabilities $p(m|\boldsymbol{x})$.

Reward: Let K denote the time step when the termination mechanism is triggered. Since the generative probability $p\big(\boldsymbol{y}|\boldsymbol{x}; \mathcal{S}_{\boldsymbol{x}}^{(K)}\big)$ can only be calculated after the termination is triggered, the reward can only be received at the termination step. We then set the log likelihood $\log p\big(\boldsymbol{y}|\boldsymbol{x}; \mathcal{S}_{\boldsymbol{x}}^{(K)}\big)$ as the reward of actions. The filter, meanwhile, outputs $\mathcal{S}_{\boldsymbol{x}}^{(K)}$ as $\mathcal{S}_{\boldsymbol{x}}$. Hence, the rewards $r_i (i = 1, 2, \cdots, K)$ are defined as:

$$\begin{aligned} r_K &= \log p\big(\boldsymbol{y}|\boldsymbol{x}; \mathcal{S}_{\boldsymbol{x}}^{(K)}\big) = \log \frac{\sum_{m \in \mathcal{S}_{\boldsymbol{x}}^{(K)}} p(m|\boldsymbol{x}) p(\boldsymbol{y}|m, \boldsymbol{x})}{\sum_{m \in \mathcal{S}_{\boldsymbol{x}}^{(K)}} p(m|\boldsymbol{x})} \\ r_k &= 0 \ (k = 1, 2, \cdots, K-1) \end{aligned} \quad (5)$$

For simplicity, we denote r_K as r.

Controlled by the policy, ERM will trigger a termination action a_t at time step K. Then it feeds the mechanism set $\mathcal{S}_{\boldsymbol{x}}^{(K)}$ as $\mathcal{S}_{\boldsymbol{x}}$ to the decoder components. Then, each mechanism embedding $\boldsymbol{m}_i \in \mathcal{S}_{\boldsymbol{x}}$ is concatenated with the context embedding $\boldsymbol{c}$ as $\tilde{\boldsymbol{c}}_i = [\boldsymbol{c}; \boldsymbol{m}_i]$. $\tilde{\boldsymbol{c}}_i$ is the mechanism-aware context embedding, and is then fed to the decoder to calcu-

late $p(\boldsymbol{y}|m_i, \boldsymbol{x})$ and generate responses with m_i:

$$p(\boldsymbol{y}|m_i, \boldsymbol{x}) = \prod_{j=1}^{|\boldsymbol{y}|} p(y_j|\boldsymbol{y}_{<j}, \tilde{\boldsymbol{c}}_i) \quad (6)$$

Here, we apply the decoder in (Cho et al. 2014) to ERM. Due to the space limitation, we omit the decoder details.

Obviously, for different $\boldsymbol{x}$ the selected $\mathcal{S}_{\boldsymbol{x}}$ may be different, and thus the amount of corresponding $\tilde{\boldsymbol{c}}_i$ may also be different. The filtering process is detailed in algorithm 1.

Training Details

In this section, we discuss how to train ERM. To train the model parameters θ, we alternately update the parameters via maximizing likelihood $p_\theta(\boldsymbol{y}|\boldsymbol{x}; \mathcal{S}_{\boldsymbol{x}})$ of all sampled states and expected reward $J(\theta)$ (detailed later).

Firstly, after the filter outputs the mechanism-aware context embeddings, we update the parameters of the decoder via maximizing likelihood $p_\theta(\boldsymbol{y}|\boldsymbol{x}; \mathcal{S}_{\boldsymbol{x}})$ for each mini-batch B, namely $\sum_{(\boldsymbol{x},\boldsymbol{y})\in B} \log p_\theta(\boldsymbol{y}|\boldsymbol{x}; \mathcal{S}_{\boldsymbol{x}})$.

Secondly, with the updated model, we obtain the log likelihood $\log p_\theta(\boldsymbol{y}|\boldsymbol{x}; \mathcal{S}_{\boldsymbol{x}})$as the reward r. The parameters are then updated again by maximizing the expected reward. Motivated by REINFORCE algorithm (Williams 1992), the expected reward for an input post $\boldsymbol{x}$ is defined as $J(\theta) = \mathbb{E}_{\mathcal{S}_{\boldsymbol{x}}, a\sim\pi}[r - b]$. Here, b is baseline reward which is the average of all rewards for each input post. Subtracting b from the rewards helps to reduce the variance in the updates (Greensmith, Bartlett, and Baxter 2004). The gradient is therefore defined as:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\mathcal{S}_{\boldsymbol{x}}, a\sim\pi_\theta}[\nabla_\theta \log \pi_\theta \cdot (r - b)] \quad (7)$$

r is the given reward and the gradient does not backpropagate through r when calculating $\nabla_\theta J(\theta)$. Obviously, the updating policy depends on the reward (likelihood) estimation. Since the parameters are randomly initialized, at this time the reward estimation is not accurate. Therefore, we maximize the expected reward once every 20 batches to make sure we could obtain a relatively accurate reward.

Note that theoretically if the post $\boldsymbol{x}$ and the response $\boldsymbol{y}$ are given, the mechanism of this post-response pair could be determined. Thus, there should be a dominant mechanism m whose probability $p(m|\boldsymbol{x}, \boldsymbol{y})$ is much larger than the others'. This will reduce the entropy $H(m|\boldsymbol{x}, \boldsymbol{y})$. Therefore we suggest using $(r - h)$ in replace of r where h is a penalized term :

$$h = \max\{\lambda, \frac{H(m|\boldsymbol{x}, \boldsymbol{y}; \mathcal{S}_{\boldsymbol{x}})}{H_{\max}(m|\boldsymbol{x}, \boldsymbol{y}; \mathcal{S}_{\boldsymbol{x}})}\}, \lambda \in [0, 1] \quad (8)$$

where $H_{\max}(m|\boldsymbol{x}, \boldsymbol{y}; \mathcal{S}_{\boldsymbol{x}}) = -\log \frac{1}{|\mathcal{S}_{\boldsymbol{x}}|}$, and we normalize the entropy H by the maximal entropy H_{max}. With this penalized term, for a sampled $\mathcal{S}_{\boldsymbol{x}}$ with lower $H(m|\boldsymbol{x}, \boldsymbol{y}; \mathcal{S}_{\boldsymbol{x}})$, it obtains higher reward.

Additionally, if the entropy ratio $\frac{H}{H_{\max}}$ is lower than λ, the penalized term will become the constant λ. This prevents the model overfitting the term. To calculate $H(m|\boldsymbol{x}, \boldsymbol{y})$, we firstly get $p(m|\boldsymbol{x}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|m,\boldsymbol{x})p(m|\boldsymbol{x})}{p(\boldsymbol{y}|\boldsymbol{x})}$ with Equ.(2), (3) and (6). Then it follows that $H(m|\boldsymbol{x}, \boldsymbol{y}; \mathcal{S}_{\boldsymbol{x}}) = -\sum_{m\in\mathcal{S}_{\boldsymbol{x}}} p(m|\boldsymbol{x}, \boldsymbol{y}) \log p(m|\boldsymbol{x}, \boldsymbol{y})$. The initial experiments shows that this track accelerates the coverage, and we set $\lambda = 0.9$ in this study.

Elastic Response Generating

With trained ERM model, we then discuss how to generate responses given a post $\boldsymbol{x}$. The main task is to determine $\mathcal{S}_{\boldsymbol{x}}$ and use $m \in \mathcal{S}_{\boldsymbol{x}}$ to respond. To determine $\mathcal{S}_{\boldsymbol{x}}$, the model first sorts the mechanisms in descending order by $p(m_i|\boldsymbol{x})$ (denoted as $m_{i_1}, m_{i_2}, \cdots, m_{i_M}$). Consistent with training, only top-K mechanisms $(m_{i_1}, m_{i_2}, \cdots, m_{i_K})$ will be selected. At generation phase, we estimate K as its expectation:

$$\hat{K} = \mathbb{E}[|\mathcal{S}_{\boldsymbol{x}}|] = \sum_{k=1}^{M} k \cdot \pi(a_t|\mathcal{S}_{\boldsymbol{x}}^{(k)}) \prod_{j=1}^{k-1} \pi(a_c|\mathcal{S}_{\boldsymbol{x}}^{(j)}) \quad (9)$$

where $\mathcal{S}_{\boldsymbol{x}}^{(k)}$ and $\mathcal{S}_{\boldsymbol{x}}^{(j)}$ contain mechanisms with top-k and top-j probability $p(m|\boldsymbol{x})$ respectively. Note that we round $\hat{K}$ to an integer when determining $\mathcal{S}_{\boldsymbol{x}}$. After $\mathcal{S}_{\boldsymbol{x}}$ is determined, for each mechanism $m_i \in \mathcal{S}_{\boldsymbol{x}}$, we generate a response $\boldsymbol{y}_i$ based on the mechanism-aware context embedding $\tilde{\boldsymbol{c}}_i = [\boldsymbol{c}; \boldsymbol{m}_i]$ via beam search. Here, $\boldsymbol{m}_i$ is mechanism embedding and is trained together with other model parameters. Note that we repeat beam search and make sure $\boldsymbol{y}_i$ is different from previously generated responses. Finally, we set the generated $\hat{K}$ responses as the final output.

Experiment Process

Dataset Details

We utilize the dataset in (Zhou et al. 2017) for experiments, which is collected from Tecent Weibo[1]. In total, there are $815,852$ pairs, among which $775,852$ are for training, and $40,000$ for model validation.

Benchmark Methods

We implemented seven types of conversation models for comparison:

1. SEQ2SEQ (Sutskever, Vinyals, and Le 2014): The RNN model that utilizes the last hidden state of the encoder as the initial hidden state of the decoder;
2. ENCDEC (Cho et al. 2014): The RNN model that feeds the last hidden state of the encoder to every cell and softmax unit of the decoder;
3. ATT (Bahdanau, Cho, and Bengio 2015): The RNN model based on EncDec with attention signal;
4. NRM (Shang, Lu, and Li 2015): Neural Responding Machine with both global and local schemes;
5. MMI-bidi and MMI-antiLM (Li et al. 2016a): The RNN model uses Maximum Mutual Information (MMI) as the objective function to reorder generated responses. Two parameters are given: $\lambda = 0.5$ and $\gamma = 1$;

[1] http://t.qq.com/?lang=en_US

Table 1: The performance of each model. The "Top-k" denotes the responses with top-k probabilities in each group. Specially, the responses of MARM-25, ERM-25 and ERM-25-All are sorted in descending order by $p(m|\boldsymbol{x})$.

Models	**%Acceptable***					**%Bad**	**%Normal**	**%Good**	**BLEU-4**	**Top-5 Diversity**	**Total Diversity**
	Top-1	Top-2	Top-3	Top-4	Top-5						
ENCDEC	31.78	34.17	37.11	38.92	40.47	59.53	35.93	4.53	8.78	0.3074	-
SEQ2SEQ	45.00	48.22	48.56	48.19	48.40	51.60	40.80	7.60	12.45	0.2806	-
ATT	47.89	49.89	51.70	53.00	53.11	46.88	40.40	12.71	13.89	0.2760	-
NRM	53.00	54.39	54.93	55.42	55.20	44.80	45.73	9.47	13.73	0.2434	-
MMI-antiLM	49.00	45.67	44.11	43.50	43.60	56.40	36.53	7.07	8.56	0.2147	-
MMI-bidi	58.67	58.33	55.89	54.67	54.60	45.40	45.07	9.53	4.41	0.3060	-
MARM-4	65.00	66.83	65.44	64.58	64.60	35.40	41.00	23.60	11.56	0.5033	-
MARM-25	58.67	54.83	53.56	53.00	51.93	48.07	28.40	23.53	5.87	0.4280	0.2942
ERM-25	70.67	72.83	71.78	71.75	71.60	29.85	44.76	25.39	12.23	0.5493	0.5308
ERM-25-All	70.67	72.83	71.78	71.75	71.60	51.24	31.12	17.64	7.49	0.5493	0.3073

6. MARM-4 and MARM-25 (Zhou et al. 2017): The RNN model uses multiple mechanisms to improve the generation quality and diversity. Here, MARM-4 utilizes the default setting: 4 mechanisms for training and mechanisms with top-2 $p(m|\boldsymbol{x})$ for responding. MARM-25 uses 25 mechanisms for training and all the mechanisms for responding (every mechanism generates one response);
7. ERM-25 and ERM-25-All: ERM model reported in this paper with 25 mechanisms for training. The only difference between ERM-25 and ERM-25-All is: ERM-25 use the elastic generating method proposed in the section *Elastic Response Generating*, while ERM-25-All uses all the mechanisms for responding (each mechanism generates one response).

Implementation Details

We use a vocabulary of 28,000 Chinese words in coarse-grained segmentation. This vocabulary covers 98.26% of the words in the training corpus. The out-of-vocabulary words are replaced with a special token "UNK".

We implement models using Theano (Theano Development Team 2016). For all the models, the dimension of the word embedding is 128, the dimension of hidden state is 1024, and one-layer RNN with GRU (Cho et al. 2014) activation function is utilized. For initialization, parameters are sampled from a uniform distribution between -0.01 and 0.01. For training, ADADELTA (Zeiler 2012; Graves 2013) is used for optimization. We stop training after the error over the validation set does not decrease for 7 consecutive epochs. For each model, the parameters with the largest likelihood on validation set are selected for final comparison. For generating responses, beam search with beam size 200 is applied.

Human Judgment

Due to the high diversity of responses in real-world corpus, it is non-trivial to construct a dataset covering all responses for each post. Hence, we employ human judges in our experiments. In detail, 3 labelers were invited to evaluate the quality of responses to 300 randomly sampled posts.

For each post, 1) ENCDEC, SEQ2SEQ, ATT, NRM, MMI-antiLM, MMI-bidi and MARM-4 generate top-5 different responses; 2) MARM-25 and ERM-25-All generate 25 different responses (every mechanism generates one response); 3) ERM-25 dynamic selects the generating mechanisms via the method in the section *Elastic Response Generating*. For fair comparison, we merge all the model responses in a single file, and shuffle the file to prevent the labelers from knowing which model a response is generated by.

Similar to (Zhou et al. 2017), for each response the labelers determine the quality to be one of the following three levels:

- **Bad**: The response is ungrammatical and irrelevant.
- **Normal**: The response is basically grammatical and relevant to the input post but trivial and dull, e.g. "Yes" "No" "I don't know".
- **Good**: The response is not only grammatical and relevant to the input post, but also meaningful and informative.

The response on Normal and Good level is "Acceptable". From labeling results, average percentages of responses in different levels are calculated. Additionally, to evaluate the diversity of the responses, for each post the labelers annotate the number of different meanings among the acceptable responses, namely n. Let K denote the number of responses generated by a given model. The diversity score is defined as $\frac{n}{K}$. We define the model's diversity score as the average diversity scores of its posts. We also report the diversity score derived from top-5 responses of each model, namely Top-5 diversity.

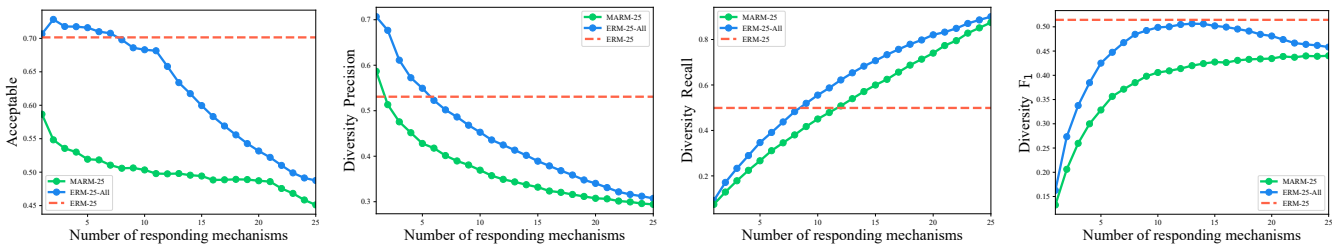
To this end, labeling agreement is evaluated by Fleiss' kappa (Fleiss 1971) which is a measure of inter-rater consistency. In this experiment, kappa value $\kappa = 0.66$ (substantial agreement). Furthermore, we report BLEU-4 (Papineni et al. 2002) scores (percentage) for these 300 posts, which is conventionally applied in translation tasks. Since some researchers indicate that BLEU may not be a good measure for dialog evaluation(Liu et al. 2016), we consider human judgment as the major measurement in the experiments.

Experimental Results and Analysis

Experimental Results

We summarize the experimental results in Table 1.

Since all the models generate at least 5 responses, we firstly observe the Top-5 Acceptable and diversity score. The



(a) Acceptable ratio. (b) Diversity precision. (c) Diversity recall. (d) Diversity F_1.

Figure 2: Comparisons between MARM-25, ERM-25 and ERM-25-All.

best baseline method MARM-4 achieves 64.60% Top-5 Acceptable ratio, while ERM-25 reaches 71.60% with an increase percentage of 10.84%. We observe that this improvement is mainly from more Normal responses are generated (41.00% vs. 46.53%), indicating that some irrelevant and ungrammatical responses are repaired to Normal in ERM. Additionally, MARM-4 achieves 50.38% Top-5 diversity, while ERM-25 reaches 54.93% with an increase percentage of 9.05%, indicating that these repaired responses might obtain different meanings from the others. Here, since ERM-25 and ERM-25-All share the Top-5 responses, they obtain the same Top-5 Acceptable and Diversity scores. We notice that ERM-25-All obtains unsatisfactory Normal and Good scores. The reason we conjecture is that we force the model to use all 25 mechanisms to generate 25 responses, instead of selecting a suitable mechanism subset to generate. We experimentally discover that mechanisms with lower $p(m|\boldsymbol{x})$ often generates lower-quality responses. These responses reduces the Acceptable, Normal and Good scores.

It is interesting to find the Acceptable score of MARM-25 is not satisfactory. While its Good score is close to MARM-4, the Top-5 acceptable (51.93%) and Normal score (28.40%) is not competitive. We conjecture that if the mechanism number is set to be large while the filter module is not applied to fine-select the responding mechanisms, the model may overfit the corpus. In other words, in these settings MARM might be learning some noise.

Furthermore, to empirically demonstrate that ERM's elastic generation method (discussed in the section *Elastic Response Generating*) outperforms the one proposed in (Zhou et al. 2017), which utilizes fixed-number mechanisms for responding every post, we calculate Acceptable rate and Diversity for ERM-25-All and MARM-25 which uses mechanisms of Top-1, Top-2, $\cdots$, Top-25 probability $p(m|\boldsymbol{x})$, and compare them to the score of ERM-25.

Motivated by the F_1 score used in Information Retrieval field, we define the Diversity-Precision P, Diversity-Recall R and Diversity-F_1 scores for measuring the quality of a generated response. For the i-th testing post, we firstly merge the responses by MARM-25, ERM-25 and ERM-25-All together. Then for these merged responses, the labelers annotate the number of different meanings in the acceptable responses, namely N_i. After this, for each model we independently collect the statistics of how many different meanings are in its acceptable responses, namely n_i. If a model generates K_i responses, we define Diversity-Precision $P = \frac{1}{L}\sum_{i=1}^{L}\frac{n_i}{K_i}$ (same as the aforementioned diversity score), Diversity-Recall $R = \frac{1}{L}\sum_{i=1}^{L}\frac{n_i}{N_i}$ and Diversity-F_1 $F_1 = \frac{2PR}{P+R}$ where L is the number of testing posts. Hence, if one model generates more relevant and grammatical responses but also more diverse responses, the Diversity-F_1 becomes higher. We believe that a higher F_1 score indicates that the model can better balance the acceptable and diversity performance.

Observed from Fig.2(a) and Fig.2(b), the red dashed line is the score of ERM-25. For Acceptable ratio, ERM-25 outperforms other two models, except ERM-25-All from Top-1 to Top-7. For Diversity-Precision, ERM-25 outperforms other models except ERM-25-All from Top-1 to Top-7, and MARM-25 Top-1. We can also find that as the number of used mechanisms increases, the Acceptable and Diversity-Precision P decreases. As we discussed before, the mechanisms with lower $p(m|\boldsymbol{x})$ may generate low-quality responses and reduce the Acceptable score. Similarly, since the Diversity-Precision is calculated based on the acceptable response, the Diversity-Precision also decreases.

The observation of Diversity-Recall R is opposite (Fig.2(c)). The red dashed line is ERM-25. ERM-25 outperforms other models when mechanisms are restricted to less or equal to Top-9. Fig.2(c) also shows R increases as the number of responding mechanisms increases. It indicates that mechanism with a lower $p(m|\boldsymbol{x})$ might generate more distinctive responses.

To summarize, we discover that as the number of responding mechanisms increases, the diversity recall increases while precision decreases. As Diversity-Precision and Diversity-Recall might not ideally represent the overall performance of a model, we consider Diversity-F_1 as the major measurement. The model with both high Diversity-Precision and high Diversity-Recall could achieve a relatively high Diversity-F_1. Observed from Fig. 2(d), the Diversity-F_1 of ERM-25 is 0.5144. It is slightly more than the maximal Diversity-F_1 of ERM-25-All (0.5064), and visibly outperforms MARM-25 (0.4403). This experimentally demonstrates that ERM's elastic generating method is preferable to always utilizing a fixed-number of mechanisms to respond posts.

Analysis on Responding Mechanisms

In this section, we investigate the distribution of the number of responding mechanisms, and explore the relationship between language styles and mechanisms.

We randomly sample 50,000 posts in the training set. After removing the spams and sentences containing "UNK", 48,876 posts are left. For each post, we use ERM model to estimate the expected number of responding mechanisms $\hat{K} = \mathbb{E}[|\mathcal{S}(\boldsymbol{x})|]$ where $\hat{K}$ is a real number (discussed in section *Elastic Response Generating*). We plot the histogram of mechanism number $\hat{K}$ for responding a given post, shown in Fig.3. Fig.3 shows that most of $\hat{K}$ are in $[7, 12]$. In our experimental corpus, it is interesting to discover: 1) A few posts with $\hat{K} \in [7, 8]$ are yes-or-no questions, e.g. "Are you 16 years old?". We believe that the model tends to use less mechanisms to respond these specific posts. 2) A number of posts with $\hat{K} \in (10, 12]$ are wh-questions. We conjecture that this post type usually has more diverse responses than others. 3) Posts with $\hat{K} \in (8, 10]$ are likely to be common sentences. For example, "Your mother called you home for dinner".

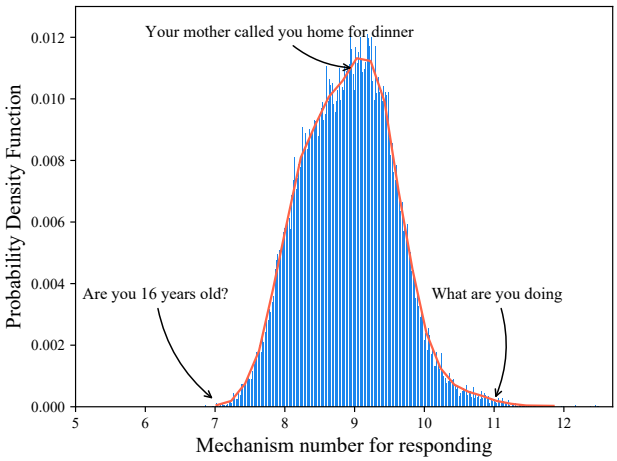


Figure 3: Histogram of mechanism number $\hat{K}$ for responding a given post. The red curve is smoothed.

Additionally, examples in Table 2 show how ERM select mechanisms and generate relevant and diverse responses. We find that some mechanisms correlate to specific language styles. For example, the responses generated by mechanism 1 are usually strong-tone sentences, and mechanism 5 usually generates questions. As ERM can automatically select suitable responding mechanism, we believe it might offer a more elegant way to control the response's language style or genre.

Related Work

End-to-end Neural Network. The basic neural-based encoder-decoder framework for generative models have been widely used in tasks such as statistical machine translation (SMT) and conversational models. In SMT, Sutskever et al. (2014) and Cho et al. (2014) used an encoder to summarize the source sentence and a decoder to generate corresponding target sentence. Later, Bahdanau et al. (2015) extended the encoder-decoder framework with the attention mechanism to improve SMT performance especially for long sentences. However, the generation diversity problem is not considered by these models. Wang et al. (2016) using extra memory cells to enhance SMT performances of RNN models. Similar to SMT, some studies show the superiority of encoder-decoder framework in conversational models. For example, Shang et al. (2015) developed a neural responding machine with global and local schemes for generating short-text conversation.

Table 2: The responses from ERM-25 for input examples

Post	我早上工作感觉死死的是怎么回事 What's wrong with me when I go to work this morning? I feel bad.	听说过外星生命吗 Do you hear alien life?	公司有人打架 Somebody come to blows in company	上班很无聊啊 It is boring to go on duty
Mechanism 1	那你要去医院看看了 You shall go to the hospital to check it.	当然有啦 Of course.	那你要去医院看看哦 You shall go to the hospital to check it.	那就好啊 That's good.
Mechanism 2	那你要去医院看看 You shall go to hospital to check it.	我是机器人 I am a robot.	为什么要骂人呀 Why you curse at other people?	你在干嘛呢 What are you doing?
Mechanism 3	你可以去网上查查啊。 You could check it through internet.	主人，快从火星上回来吧 Master, please come back from the Mars soon.	我不明白你要说什么。 I don't know what you talk about.	那就聊点有意义的事吧。 Let's talk something meaningful.
Mechanism 4	你继续把话说完啊。 Go on, tell what you want to tell.	一直都是浮云 Nothing matters.	请文明用语 Please use polite language.	好好上班吧 Work hard.
Mechanism 5	你要去医院么？ Do you go to the hospital?	你在说谁呀？ Who are you talking about?	你为什么要打人？ Why you fight with other one?	你在干嘛 What are you doing?
Mechanism 6	你在干嘛呢呀 What are you doing?	当然没有 Of course not.	怎么可能 How is it possible?	还可以吧 It's ok.
Mechanism 7	你去过什么事情？ What happen?	当然有 Of course.	为什么要打人呀 Why you fight with other one?!	聊什么啊 What are you chatting about?
Mechanism 8	夸奖 Praise	机器人都是浮云 Robot is nothing.	为什么要打人 Why you fight with other one?	无聊就来找我聊天啊。 You could chat with me if you feel borine.
Mechanism 9	那你要去医院看看哦 You shall go to hospital to check it.	我可没去过 I have not been to there.	这可不是个好主意哦。 It's not a good idea.	那你在干嘛呢 What are you doing?

Recently, researchers began to develop models for multiple-round conversation. Serban et al. (2015) extend the short-text conversational models to hierarchical neural network which has the ability to handle multiple-round conversations. Meanwhile, Sordoni et al. (2015) applied a similar hierarchical RNN model for query suggestion. Li et al. (2016b) proposed a reinforcement dialog generation model to generate informative, coherent, and easy-to-answer responses. Note that its reinforcement module is not designed for controlling the responding mechanisms. These multiple-round conversations mainly extend the encoder from only handling a single post to handling several context sentences.

Some other models are proposed to tackle response diversity problem. Li et al. (2016a) proposed the Maximum Mutual Information (MMI) as the objective to improve the diversity. Our experiments show that it decreases the acceptable ratio. Zhou et al. (Zhou et al. 2017) applied a quantitative study on the diversity problem. They then proposed MARM to generate diverse responses with different mechanisms. However, its mechanism number for responding needs to be handcrafted and might not be satisfactory for every post. Zhao et al. (2017) proposed CAVE and kgCAVE using conditional variational autoencoders. However, CAVE may be difficult to explicitly control responding mechanism, and kgCAVE needs extra feature engineering of discourse and dialog act during training which limits its applications in real-world corpus.

Sparse Modeling. For ERM, using only the selected mechanism to respond is similar to setting the $p(m|\boldsymbol{x})$ of

unselected mechanism to 0. We argue that this could be regarded as a sparse operation. Here, some studies focus on making the probability of latent variable sparse. Zhu et al. (2011) propose a sparse topic model (STC) for discovering latent representations of large collections of data. Zhang et al. (2013) proposed the sparse relational topic model (SRTM) which controls the sparsity via a sparsity-inducing regularizer. Zhang et al. (2015) applied the STC model to monitor temporal evolution of market competition. Above models add sparsity regularizers to the objective function. However, only making $p(m|\boldsymbol{x})$ sparse may not efficiently determine whether a mechanism should be selected for responding or not, comparing to directly selecting and setting some $p(m|\boldsymbol{x})$ to 0.

Conclusion and Future Work

In this study, we proposed an additional filter component to the encoder-diverter-decoder structure, aiming to explicitly train a conversational model over large set of mechanisms and explicitly model which mechanisms are suitable for a given input post. We empirically demonstrate that the proposed model can generate more acceptable and diverse responses comparing to the baseline methods. It also offers possibility to automatically learn and control response language style in future work.

The work reported in this paper can be regarded as a part of a larger blueprint: developing techniques for end-to-end generative model, such as dialog system, image captioning and other applications. For data set which implicitly contains 1-to-n mapping relations, we suggest injecting ERM into the conventional models to boost their performances. This will be the focus of our future work in this area.

Acknowledgments

This work was supported by the National Key Research and Development Program of China under Grant No. 2017YFB1002104, the National Natural Science Foundation of China (No.61473274, 61573335).

This work was also supported by WeChat Tencent. We thank Leyu Lin, Lixin Zhang, Cheng Niu and Xiaohu Cheng for their constructive advices. We also thank the anonymous AAAI reviewers for their helpful feedback.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. *ICLR*.

Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP*.

Fleiss, J. L. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*.

Graves, A. 2013. Generating sequences with recurrent neural networks. *arXiv*.

Greensmith, E.; Bartlett, P. L.; and Baxter, J. 2004. Variance reduction techniques for gradient estimates in reinforcement learning. *JMLR*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. In *Neural Computation*.

Li, J.; Galley, M.; Brockett, C.; Gao, J.; and Dolan, W. B. 2016a. A diversity-promoting objective function for neural conversation models. In *NAACL-HLT*.

Li, J.; Monroe, W.; Ritter, A.; Galley, M.; Gao, J.; and Jurafsky, D. 2016b. Deep reinforcement learning for dialogue generation. In *EMNLP*.

Liu, C.-W.; Lowe, R.; Serban, I. V.; Noseworthy, M.; Charlin, L.; and Pineau, J. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *ACL*.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Serban, I. V.; Sordoni, A.; Bengio, Y.; Courville, A.; and Pineau, J. 2015. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*.

Shang, L.; Lu, Z.; and Li, H. 2015. Neural responding machine for short-text conversation. In *ACL*.

Sordoni, A.; Bengio, Y.; Vahabi, H.; Lioma, C.; Simonsen, J. G.; and Nie, J.-Y. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *CIKM*.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv*.

Wang, M.; Lu, Z.; Li, H.; and Liu, Q. 2016. Memory-enhanced decoder for neural machine translation. In *EMNLP*.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*.

Wiseman, S., and Rush, A. M. 2016. Sequence-to-sequence learning as beam-search optimization. In *EMNLP*.

Zeiler, M. D. 2012. ADADELTA: An Adaptive Learning Rate Method. *arXiv*.

Zhang, H.; Kim, G.; and Xing, E. P. 2015. Dynamic topic modeling for monitoring market competition from online text and image data. In *ACM SIGKDD*.

Zhang, A.; Zhu, J.; and Zhang, B. 2013. Sparse online topic models. In *WWW*.

Zhao, T.; Zhao, R.; and Eskenazi, M. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. *ACL*.

Zhou, G.; Luo, P.; Cao, R.; Lin, F.; Chen, B.; and He, Q. 2017. Mechanism-aware neural machine for dialogue response generation. In *AAAI*.

Zhu, J., and Xing, E. P. 2011. Sparse topical coding. In *UAI*.