Vocabulary Pyramid Network: Multi-Pass Encoding and Decoding with Multi-Level Vocabularies for Response Generation

Cao Liu[1,2], **Shizhu He**[1], **Kang Liu**[1,2], **Jun Zhao**[1,2]
[1] National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, 100190, China
[2] University of Chinese Academy of Sciences, Beijing, 100049, China
{cao.liu, shizhu.he, kliu, jzhao}@nlpr.ia.ac.cn

Abstract

We study the task of response generation. Conventional methods employ a fixed vocabulary and one-pass decoding, which not only make them prone to safe and general responses but also lack further refining to the first generated raw sequence. To tackle the above two problems, we present a Vocabulary Pyramid Network (VPN) which is able to incorporate multi-pass encoding and decoding with multi-level vocabularies into response generation. Specifically, the dialogue input and output are represented by multi-level vocabularies which are obtained from hierarchical clustering of raw words. Then, multi-pass encoding and decoding are conducted on the multi-level vocabularies. Since VPN is able to leverage rich encoding and decoding information with multi-level vocabularies, it has the potential to generate better responses. Experiments on English Twitter and Chinese Weibo datasets demonstrate that VPN remarkably outperforms strong baselines.

1 Introduction

As one of the long-term goals in AI and NLP, automatic conversation devotes to constructing automatic dialogue systems to communicate with humans (Turing, 1950). Benefited from large-scale human-human conversation data available on the Internet, data-driven dialog systems have attracted increasing attention of both academia and industry (Ritter et al., 2011; Shang et al., 2015a; Vinyals and Le, 2015; Li et al., 2016a,c, 2017).

Recently, a popular approach to build dialog engines is to learn a response generation model within an encoder-decoder framework such as sequence-to-sequence (Seq2Seq) model (Cho et al., 2014a). In such a framework, an encoder transforms the source sequence into hidden vectors, and a decoder generates the targeted sequence based on the encoded vectors and previously generated words. In this process, the encoder and decoder share a vocabulary (word list)[1], and the targeted words are typically performed by a *softmax* classifier over the vocabulary word-by-word.

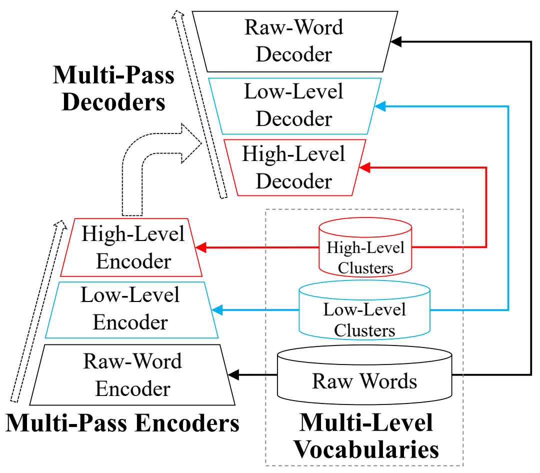

Figure 1: Vocabulary pyramid networks for response generation. The dialogue input (context) and output (response) are represented by multi-level *vocabularies* (e.g., raw words, low-level clusters and high-level clusters) and then processed by multi-pass encoder and decoder.

However, such typical Seq2Seq model is prone to generate safe and repeated responses, such as "*Me too*" and "*I don't know*". In addition to the *exposure bias* issue[2], the main reasons of this problem include: 1) a fixed (single) vocabulary (word list) in decoding, which usually covers high-frequency words, so it is easy to capture high-frequency patterns (e.g., "*Me too*") and lose a great deal of content information in middle and low-frequency patterns; 2) one-pass decoding,

[1]Encoder and decoder may have different word lists. We find it performs closely using same or different vocabularies.
[2]A model generates the next word given the previous gold words in training while it is based on previously predicted words in the test (Ranzato et al., 2016).

where word-by-word generation from left to right is prone to error accumulation since previously generated erroneous words will greatly affect future un-generated words. More importantly, it can leverage only the previously generated words but not the future un-generated words.

In fact, there are some researches in text generation tasks such as dialogue generation, machine translation and text summarization, are dedicated to solving the above issues. In order to alleviate issues on the fixed vocabulary, Wu et al. (2018a) incorporated dynamic vocabulary mechanism into Seq2Seq models, which dynamically allocates vocabularies for each input by a vocabulary prediction model. Xing et al. (2017) presented topic aware response generation by incorporating topic words obtained from a pre-trained LDA model (Blei et al., 2003). Besides, several works attempted to solve the dilemma of one-pass decoding. Xia et al. (2017) proposed deliberation network for sequence generation, where the first-pass decoder generates a rough sequence and then the second-pass decoder refines the rough sequence.

However, so far there has been no unified framework to solve both of the aforementioned problems. In this study, we present Vocabulary Pyramid Networks (VPN) to tackle the issues of one fixed vocabulary and one-pass decoding simultaneously. Specifically, VPN incorporates multi-pass encoding and decoding with multi-level vocabularies into response generation. As depicted in Figure 1, the multi-level vocabularies contain raw words, low-level clusters and high-level clusters, where low-level and high-level clusters are obtained from hierarchical clustering of raw words. Afterward, the multi-pass encoder (raw-word encoder, low-level encoder, and high-level encoder) gradually works on diminishing vocabularies from raw words to low-level clusters until to high-level clusters, and it looks like a "*pyramid*" concerning the vocabulary size. On the other side, the multi-pass decoder gradually increases the size of processed vocabularies from high-level clusters to low-level clusters and finally to raw words.

From a theoretical point of view, people usually associate raw input words with low-level or high-level abstractions like semantic meanings and concepts on human-human conversations. Based on the abstractive cognition, people organize contents and select the expressive words as the response (Xing et al., 2017). From a practical perspective, VPN is able to capture much more sequence information with multi-level vocabularies. As a result, VPN has the potential to generate better responses.

To verify the effectiveness of the proposed model, we conduct experiments on two public response generation datasets: English *Twitter* and Chinese *Weibo*. Both automatic and manual evaluations demonstrate that the proposed VPN is remarkably better than the state-of-the-art.

2 Background

2.1 Sequence-to-Sequence Model

In Seq2Seq models (Cho et al., 2014a), an encoding RNN (recurrent neural network) transforms the source sequence $X = \{x_1, x_2, ..., x_{L_X}\}$ into distributed representations $\mathbf{H} = \{\mathbf{h}_1, \mathbf{h}_1, ..., \mathbf{h}_{L_X}\}$ through a basic model: $\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1})$. Here, $\mathbf{x}_t$ is the word embedding for x_t, f is a non-linear transformation, where GRU (Cho et al., 2014b) and LSTM (Hochreiter and Schmidhuber, 1997) are widely used for capturing long-term dependencies. Then a decoder generates the targeted sequence $Y = \{y_1, y_2, ..., y_{L_Y}\}$ as follows:

$$\mathbf{s}_t = f([\mathbf{y}_{t-1}, \mathbf{c}], \mathbf{s}_{t-1}) \tag{1}$$

$$p(y_t|y_{<t}, X) = g(\mathbf{y}_{t-1}, \mathbf{c}, \mathbf{s}_t) \tag{2}$$

where $\mathbf{c} = \mathbf{h}_{L_X}$, $\mathbf{s}_t$ is the decoding state in time step t, and g is a non-linear function. In the basic Seq2Seq models, each word is generated from a same context vector $\mathbf{c}$. In order to capture different contexts for each generated word, attention mechanism (Bahdanau et al., 2015) extracts dynamic context vector $\mathbf{c}_t$ in different decoding time steps. Formally, $\mathbf{c}_t = \sum_{j=1}^{L_X} \alpha_{ij}\mathbf{h}_j$, $\alpha_{ij} \propto exp(\eta(\mathbf{s}_{i-1}, \mathbf{h}_j))$, where η is a non-linear function.

2.2 Deliberation Network

Conventional Seq2Seq models can leverage only the generated words but not the un-generated words in decoding, so they lack global information to *refine* and *polish* the raw generated sequence. The deliberation network (Xia et al., 2017) is proposed to deal with this issue. A deliberation network has two decoders, where the first-pass decoder generates a raw word sequence $Y^1 = \{y_1^1, y_2^1, ..., y_{L_{Y^1}}^1\}$ and the second-pass decoder polishes the raw word sequence. In the second-pass decoder, an extra attention model is leveraged to selectively read the output vector sequence Y^1 from the first-pass decoder, and then generate the refined output sequence $Y^2 = \{y_1^2, y_2^2, ..., y_{L_{Y^2}}^2\}$.

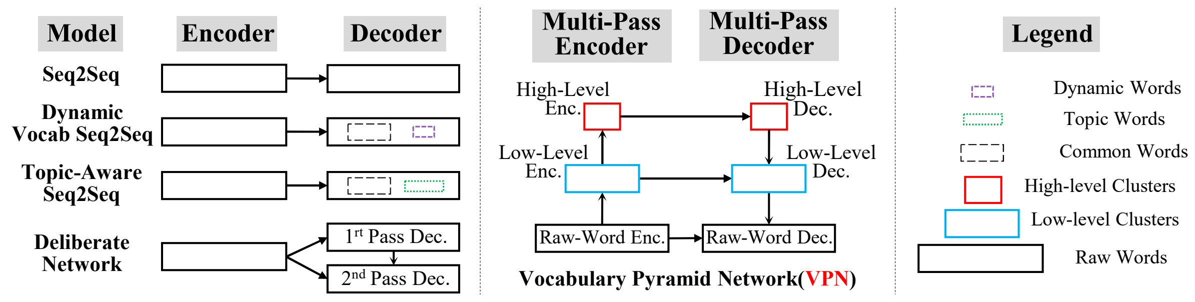

Figure 2: Differences in our VPN with typical Seq2Seq model and its variations, where different rectangles denote different vocabularies (details in "Legend"). *Seq2Seq* uses a vocabulary (word list) in decoding. *Dynamic vocabulary Seq2Seq* integrates a common vocabulary and a dynamic vocabulary in decoding. *Topic-Aware Seq2Seq* incorporates topic words for each input. *Deliberate network* exploits first-pass and two-pass decoder within the same vocabulary list. VPN employs multi-pass encoder and multi-pass decoder with multi-level vocabularies (raw words, low-level clusters and high-level cluster). Among these models, only VPN makes use of vocabularies beyond words. Therefore, VPN could capture rich encoding and decoding information with multi-level vocabularies.

3 Methodology

3.1 Model Overview

As shown in Figure 2, VPN consists of three submodules: multi-level vocabularies (Section 3.2), multi-pass encoder (Section 3.3) and multi-pass decoder (Section 3.4). Specifically, multi-level vocabularies contain raw words, low-level clusters and high-level clusters (black, blue and red solid rectangles in Figure 2). The multi-pass encoder starts from the raw words and then to the low-level clusters finally to the high-level clusters. In contrast, the multi-pass decoder works from the high-level clusters to the low-level clusters until to the raw words. The details of each component are in the following.

3.2 Multi-Level Vocabularies

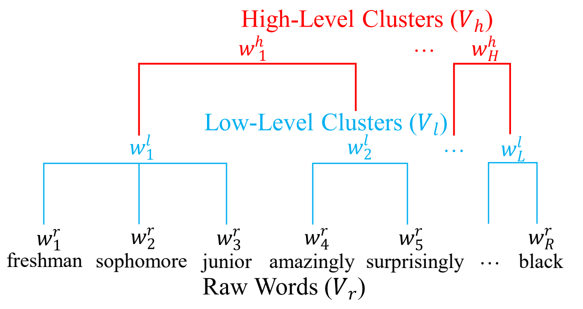

Figure 3: Multi-level vocabularies via hierarchical clustering.

As illustrated in Figure 3, multi-level vocabularies contain three different vocabularies: raw words, low-level clusters and high-level clusters. Specifically, the raw words are the original words in the training data, and they are denoted as $V_r = \{w_1^r, w_2^r, ..., w_R^r\}$. The raw words are agglomerated into low-level clusters $V_l = \{w_1^l, w_2^l, ..., w_L^l\}$ and high-level clusters $V_h = \{w_1^h, w_2^h, ..., w_H^h\}$ by "*bottom-up*" hierarchical clustering. In order to decide which clusters could be agglomerated, we utilize the implementation of hierarchical clustering in *Scipy*[3]. Specifically, we pre-train raw-word embeddings by the word2vec model[4] as inputs, and then we leverage the Ward (Ward, 1963) linkage and maxclust[5] criterion to automatically construct hierarchical clustering.

In this way, we could obtain three different vocabularies: V_r, V_l and V_h, where their vocabulary sizes are decreased: $|V_r|>|V_l|>|V_h|$, and it looks like a "*pyramid*" concerning the vocabulary size. It should be emphasized that an original input sequence could be expanded into three input sequences through the three vocabulary lists, and it is the same for the output sequence.

3.3 Multi-Pass Encoder

The encoder aims to transform input sequences into distributional representations. In order to capture much more information from different input sequences, VPN employs a multi-pass encoder, which contains three different encoders in order: raw-word encoder, low-level encoder and high-level encoder. As a result, the multi-pass encoder is able to encode more and more abstractive infor-

[3]https://scipy.org/

[4]Implemented in https://radimrehurek.com/gensim/models/word2vec.html

[5]https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.cluster.hierarchy.fcluster.html

mation from words to clusters. The details are in the following.

Raw-Word Encoder

The raw-word encoder accepts an input sequence of word ids from raw words V_r. A bi-directional LSTM (Schuster and Paliwal, 1997) is leveraged to capture the long-term dependency from forward and backward directions. The concatenated representation of bi-directional hidden states: $\mathbf{h}_i^r = [\overrightarrow{\mathbf{h}}_i^r, \overleftarrow{\mathbf{h}}_{L_i-i+1}^r]$, is regarded as the encoded vector for each input word. Finally, the input sequence is transformed into a hidden state sequence:

$$\mathbf{H}^r = \{\mathbf{h}_1^r, \mathbf{h}_2^r, ..., \mathbf{h}_{L_i}^r\} \tag{3}$$

Specifically, the initiated hidden state is a zero vector, and the hidden state ($\mathbf{h}_{L_i}^r$) in the last word could be used for initiating the next encoder (low-level encoder).

Low-Level Encoder

Low-level encoder is similar to the raw-word encoder. However, low-level encoder takes a sequence of low-level cluster ids from V_l as inputs, and the hidden state is initiated by the last hidden state of the raw-word encoder ($\mathbf{h}_{L_i}^r$). Similarly, we can obtain the hidden state sequence in the low-level encoder:

$$\mathbf{H}^l = \{\mathbf{h}_1^l, \mathbf{h}_2^l, ..., \mathbf{h}_{L_i}^l\} \tag{4}$$

High-Level Encoder

The high-level encoder accepts a sequence of high-level cluster ids from V_h, and the initiated hidden state is the final hidden state $\mathbf{h}_{L_i}^l$ in the low-level encoder. Finally, the hidden state sequence in the high-level encoder is denoted as follows:

$$\mathbf{H}^h = \{\mathbf{h}_1^h, \mathbf{h}_2^h, ..., \mathbf{h}_{L_i}^h\} \tag{5}$$

3.4 Multi-Pass Decoder

The decoder is responsible for generating targeted sequences. Inspired from the deliberation network (Xia et al., 2017), we present a multi-pass decoder which consists of three decoders in order: high-level decoder, low-level decoder and raw-word decoder. The three decoders have their own targeted sequences from different vocabulary lists, and the multi-pass decoder first generates the abstractive (high and low-level) clusters and then generates the raw (specific) words. It is different from the deliberation network where both the first-pass decoder and the second-pass decoder aim to generate raw words in the same vocabulary. The details of our multi-pass decoder are in the following.

High-Level Decoder

The high-level decoder generates a high-level cluster sequence from V_h. Similar to human-human conversations, where people usually associate an input message with high-level abstractions like concepts in their minds before speaking, the high-level decoder generates the most abstractive cluster sequence before selecting specific words as responses.

The high-level decoder is based on another LSTM, which is initiated with the last hidden state $\mathbf{h}_{L_i}^h$ in the high-level encoder. In order to decide which parts of sources need more attention, an attention mechanism (Bahdanau et al., 2015) is introduced in the high-level decoder. Intuitively, the encoded hidden state sequence $\mathbf{H}^h$ in the high-level encoder contains the most relevant encoded information for the high-level decoder because they share the same vocabulary V_h. Nevertheless, in order to capture much more encoded information from the source sequences, the high-level decoder adopts three attention models to attentively read different encoded state sequences: $\mathbf{H}^r$, $\mathbf{H}^l$ and $\mathbf{H}^h$ (Equation 3-5), respectively. Take $\mathbf{H}^r$ as an example, at each decoding time step j, the high-level decoder dynamically chooses the context vector $\mathbf{c}_j^{hr}$ based on $\mathbf{H}^r = \{\mathbf{h}_1^r, \mathbf{h}_2^r, ..., \mathbf{h}_{L_i}^r\}$ and the decoding state $\mathbf{s}_{j-1}^h$ as follows:

$$\mathbf{c}_j^{hr} = \sum\nolimits_{i=1}^{L_i} \alpha^{ji}\mathbf{h}_i^r; \;\; \alpha^{ji} = \frac{e^{\rho(\mathbf{s}_{j-1}^h, \mathbf{h}_i^r)}}{\sum_{i'} e^{\rho(\mathbf{s}_{j-1}^h, \mathbf{h}_{i'}^r)}} \tag{6}$$

where ρ is a non-linear function to compute the attentive strength. Similarly, the attentive context vectors ($\mathbf{c}_j^{hl}$ and $\mathbf{c}_j^{hh}$) from the low-level and high-level encoders could be calculated by the attention models. Based on $\mathbf{c}_j^{hr}$, $\mathbf{c}_j^{hl}$ and $\mathbf{c}_j^{hh}$, the decoding state $\mathbf{s}_j^h$ is updated as:

$$\mathbf{s}_j^h = f^h([\mathbf{y}_{j-1}^h, \mathbf{c}_j^{hr}, \mathbf{c}_j^{hl}, \mathbf{c}_j^{hh}], \mathbf{s}_{j-1}^h) \tag{7}$$

where $\mathbf{y}_{j-1}^h$ is the embedding vector of the previously decoded cluster at time step $j-1$, and f^h is the decoding LSTM unit. Finally, the targeted cluster is typically obtained by a *softmax* classifier over V_h based on the embedding similarity. In this way, the high-level decoder could generate the output sequence $\mathbf{y}^h = \{y_1^h, y_2^h, ..., y_{L_o}^h\}$, which corresponds to the embedding sequence:

$$\mathbf{Y}^h = \{\mathbf{y}_1^h, \mathbf{y}_2^h, ..., \mathbf{y}_{L_o}^h\} \tag{8}$$

Low-Level Decoder

Once the high-level cluster sequence is generated from the high-level decoder, it could be leveraged to the low-level decoder for further decoding the low-level cluster sequence. Based on the three encoded state sequences ($\mathbf{H}^r, \mathbf{H}^l, \mathbf{H}^h$) and the output embedding sequence $\mathbf{Y}^h$ of the high-level decoder, the low-level encoder generates another sequence from the low-level clusters V_l.

The low-level decoder is similar to the high-level decoder. However, there still are some differences between them: 1) The initiated hidden state $\mathbf{s}_0^l$ in the low-level decoder is performed as the final decoding state $\mathbf{s}_{L_o}^h$ in the high-level decoder. 2) The attentive context vectors ($\mathbf{c}_j^{lr}, \mathbf{c}_j^{ll}$ and $\mathbf{c}_j^{lh}$) from encoded state sequences are calculated with different parameters compared to ones in the high-level decoder. 3) Inspired from deliberation networks, previously generated sequence $\mathbf{Y}^h$ in the high-level encoder is fed into the low-level decoder, where high-level (global) information guides low-level generations, and another attention model is leveraged to capture such information, which is similar to Equation 6 mathematically:

$$\mathbf{o}_j^{lh} = \sum\nolimits_{i=1}^{L_o} \beta^{ji} \mathbf{y}_i^h \tag{9}$$

where the attentive weight β^{ji} is calculated from the low-level decoding states $\mathbf{s}_{j-1}^l$ and output embedding sequence $\mathbf{Y}^h$ (Equation 8) in the high-level decoder. Thereafter, $\mathbf{o}_j^{lh}$ is concatenated to update the decoded hidden state as follows:

$$\mathbf{s}_j^l = f^l([\mathbf{y}_{j-1}^l, \mathbf{c}_j^{lr}, \mathbf{c}_j^{ll}, \mathbf{c}_j^{lh}, \mathbf{o}_j^{lh}], \mathbf{s}_{j-1}^l) \tag{10}$$

where f^l is another LSTM unit. Finally, the output y_j^l is generated by a *softmax* classifier from V_l based on embedding similarity.

Raw-Word Decoder

After obtaining the high-level and low-level cluster sequence, the next step is to produce the final raw word sequence from V_r by the raw-word decoder. The hidden state of the raw-word decoder $\mathbf{s}_0^h$ is initiated with the final decoding state $\mathbf{s}_{L_o}^l$ in the low-level decoder. The decoding state in the raw-word decoder is updated as follows:

$$\mathbf{s}_j^r = f^r([\mathbf{y}_{j-1}^r, \mathbf{c}_j^{rr}, \mathbf{c}_j^{rl}, \mathbf{c}_j^{rh}, \mathbf{o}_j^{rl}, \mathbf{o}_j^{rh}], \mathbf{s}_{j-1}^r) \tag{11}$$

where $\mathbf{c}_j^{rr}, \mathbf{c}_j^{rl}, \mathbf{c}_j^{rh}$ are attentive context vectors from three encoded hidden state sequences. $\mathbf{o}_j^{rh}$ and $\mathbf{o}_j^{rl}$ (similar to Equation 9) are the weighted sums of output embedding sequences from the high-level decoder and low-level decoder. Similarly, the targeted word is typically predicted by a *softmax* classifier over V_r based on the word embedding similarity. Eventually, the raw-word decoder iteratively generates a targeted word sequence $\mathbf{y}^r = \{y_1^r, y_2^r, ..., y_{L_o}^r\}$.

3.5 Learning

Multi-level vocabularies of hierarchical clustering are obtained in advance through an un-supervised way, while the multi-level encoder and decoder could be optimized with supervised learning. The encoder and decoder are totally differential, so they are able to be optimized in an end-to-end manner by the back propagation. Giving a source input and a targeted output, there are three input-output pairs obtained from different vocabulary lists: $\{\mathbf{x}^n, \mathbf{y}^n\}_{n \in \{r,l,h\}}$. Each output sequence corresponds to a training loss, and the total losses perform as follows:

$$\begin{aligned}
\mathcal{L} &= \mathcal{L}^h + \mathcal{L}^l + \mathcal{L}^r \\
\mathcal{L}^h &= \frac{-1}{L_o} \sum\nolimits_{j=1}^{L_o} \log[p(y_j^h | y_{<j}^h, \mathbf{x}^r, \mathbf{x}^l, \mathbf{x}^h)] \\
\mathcal{L}^l &= \frac{-1}{L_o} \sum\nolimits_{j=1}^{L_o} \log[p(y_j^l | y_{<j}^l, \mathbf{x}^r, \mathbf{x}^l, \mathbf{x}^h, \mathbf{Y}^h)] \\
\mathcal{L}^r &= \frac{-1}{L_o} \sum\nolimits_{j=1}^{L_o} \log[p(y_j^r | y_{<j}^r, \mathbf{x}^r, \mathbf{x}^l, \mathbf{x}^h, \mathbf{Y}^h, \mathbf{Y}^l)]
\end{aligned} \tag{12}$$

where the three negative log-likelihoods ($\mathcal{L}^h$, $\mathcal{L}^l$ and $\mathcal{L}^r$) are losses for different-level targeted outputs. $\mathbf{Y}^h$ and $\mathbf{Y}^l$ are output embedding sequences in the high-level decoder and low-level decoder, respectively. Finally, the sum of different losses in three decoders is considered as the total losses $\mathcal{L}$.

4 Experiment

4.1 Datasets

There are large-scale message-response pairs on social websites, which consist of informational text from different topics (Chen et al., 2017). Our experimental data comes from two public corpus: English "*Twitter*"[6] and Chinese "*Weibo*" (Shang et al., 2015b). In order to improve the quality of datasets, some noisy message-response pairs are filtered (e.g., containing too many punctuations or emoticons), and the datasets are randomly split into Train/Dev/Test by a proportion (9:0.5:0.5).

[6]https://github.com/Marsan-Ma-zz/chat_corpus

4.2 Implementation Details

In order to make our model comparable with typical existing methods, we keep the same experimental parameters for VPN and comparative methods. We set the vocabulary size of raw words as 34000, and the word vector dimension is 300. Moreover, source inputs are encoded by 600-dimensional vectors with bi-direction LSTMs, and responses are also decoded by LSTM with 600 dimensions. The total losses are minimized by an Adam optimizer (Kingma and Ba, 2015) with 0.0001 learning rate. Particularly, the size of low-level clusters and high-level clusters are 3400 and 340, respectively, which are significantly smaller than the size of raw words (34000), and these clusters are also represented by 300-dimensional vectors. Finally, we implemented all models with the TensorFlow.

4.3 Evaluation Metrics

Evaluation for generative responses is a challenging and under-researching problem (Novikova et al., 2017). Similar to (Li et al., 2016b; Gu et al., 2016), we borrow two well-established automatic evaluation metrics from machine translation and text summarization: BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004)[7], which could be leveraged to analyze the co-occurrences of n-gram between the generated responses and references.

In addition to automatic evaluations, we also leverage manual evaluations to enhance the evaluations. Following previous studies (He et al., 2017; Qian et al., 2018; Liu et al., 2018), we employ three metrics for manual evaluations as follows. 1) *Fluency* (*Flu.*): measuring the grammaticality and fluency of generated responses, where too short responses are regarded as lack of fluency. 2) *Consistency* (*Con.*): measuring whether the generated responses are consistent with the inputs or not. 3) *Informativeness* (*Inf.*): measuring whether the response provides informative (knowledgeable) contents or not.

4.4 Overall Comparisons

Comparison Settings. We compare VPN with the following methods:

[7]Implemented in https://github.com/Maluuba-/nlg-eval. Evaluations on Twitter are based on token level. In particular, the BLEU and ROUGE on Weibo dataset are based on the Chinese character because Chinese characters are with semantics.

Models	*Twitter*		*Weibo*	
	BLEU	ROUGE	BLEU	ROUGE
S2SA (2014)	6.12	6.42	8.95	9.06
S2STA (2017)	7.73	7.57	11.45	11.29
S2SDV (2018b)	5.91	5.87	9.05	8.71
DelNet (2017)	6.42	6.76	10.04	10.03
VPN (ours)	**8.58**	**7.88**	**12.51**	**11.76**

Table 1: Overall performance on Twitter and Weibo datasets. Note that the first three lines are only one-pass decoding, and the fourth line (DelNet) is beyond one-pass decoding.

(1) S2SA: Sequence-to-Sequence (Sutskever et al., 2014) with attention mechanisms (Bahdanau et al., 2015).

(2) S2SDV: Seq2Seq with dynamic vocabulary, the implementation is similar to Wu et al. (2018b).

(3) S2STA: Seq2Seq with topic aware networks, the implementation is similar to Xing et al. (2017). S2STA could be regarded as using dynamic vocabulary because topic words are changed along with the input.

(4) DelNet: deliberation networks, the implementation is similar to Xia et al. (2017). Different from the above methods, deliberation networks are beyond one-pass decoding.

Comparison Results. We first report overall performances on Table 1. These results support the following statements:

(1) Our VPN achieves the highest performances on English *Twitter* and Chinese *Weibo* dataset in all metrics, which demonstrates multi-pass encoding and decoding with multi-level vocabularies are able to deliver better responses than baselines.

(2) For the one-pass decoding (the first three methods in Table 1), S2STA performs the best. Pre-trained topic words for each input are able to make the generation more target-focused in S2STA. Nevertheless, it is still worse than VPN.

(3) As for models beyond one-pass decoding (the last two lines in Table 1), VPN is remarkably better than the deliberation network (DelNet), which indicates the effectiveness of multi-pass encoder and decoder with multi-level vocabularies.

4.5 The Effectiveness of Multi-Level Vocabularies

Comparison Settings. To validate the effectiveness of multi-level vocabularies obtained from hierarchical clustering, we design experiments on whether using Multi-level Vocabularies (MVs) or not. The comparison setting is shown in the first

Models	*Twitter*		*Weibo*	
	BLEU	ROUGE	BLEU	ROUGE
enc3-dec1 (SV)	6.27	6.29	6.61	7.08
enc3-dec1 (MVs)	**7.16**	**8.01**	**9.15**	**10.63**
enc1-dec3 (SV)	**7.43**	7.54	9.92	10.24
enc1-dec3 (MVs)	6.75	**7.78**	**12.01**	**10.86**
enc3-dec3 (SV)	7.44	7.56	9.95	9.70
enc3-dec3 (MVs)	**8.58**	**7.88**	**12.51**	**11.76**

Table 2: Performances on whether using multi-level vocabularies or not, where "SV" represents single vocabulary (from raw words), and "MVs" means multi-level vocabularies obtained from hierarchical clustering. "enc" and "dec" denote encoder and decoder, respectively, and numbers after them represent how many passes. For example, "enc1-dec3" means a encoder along with three passes of decoders.

column in Table 2, where numbers after "enc/dec" represent the number of encoders/decoders. "SV" denotes single vocabulary (from raw words) while "MVs" means multi-level vocabularies obtained from hierarchical clustering.

Comparison Results. Table 2 demonstrates performances on whether using multi-level vocabularies. We can observe that incorporating multi-level vocabularies could improve performances on almost all of the metrics. For example, "enc3-dec3 (MVs)" improves relative performance up to 25.73% in BLEU score compared with "enc3-dec3 (SV)" on the *Weibo* dataset. Only on the *Twitter* dataset, "enc1-dec3 (MVs)" is slightly worse than "enc1-dec3 (SV)" in the BLEU score.

4.6 The Effectiveness of Multi-Pass Encoding and Decoding

Models	*Twitter*		*Weibo*	
	BLEU	ROUGE	BLEU	ROUGE
VPN	**8.58**	**7.88**	**12.51**	**11.76**
w/o low-level ED	7.83	7.57	11.96	11.60
w/o high-level ED	6.84	7.81	10.06	10.70
w/o low&high-level ED	6.12	6.42	8.95	9.06

Table 3: Influences of multi-pass encoding and decoding, where "w/o" indicates without, "ED" represents encoder and decoder. For example, "w/o low-level ED" means removing low-level encoder and low-level decoder.

Comparison Settings. In order to demonstrate the effectiveness of multi-pass encoder and multi-pass decoder, we design an ablation study as follows. 1) w/o low-level ED: without low-level encoder and low-level decoder; 2) w/o high-level ED: without high-level encoder and high-level decoder; 3) w/o low&high-level ED: without low-level encoder/decoder and high-level encoder/decoder, which is the same as the Seq2Seq model with attention mechanisms.

Comparison Results. Results of the ablation study are shown in Table 3. We can clearly see that removing any encoder and decoder causes obvious performance degradation. Specifically, "w/o high-level ED" obtains worse performances than "w/o low-level ED". We guess that the high-level encoder and decoder are well trained since they have the smallest vocabulary (the size of high-level clusters is only 340), so removing the well-trained component ("w/o high-level ED") performs poorly (Details in Section 4.8). Furthermore, "w/o low&high-level ED" performs the worst. This further indicates that multi-pass encoder and decoder contribute to generating better responses.

4.7 Manual Evaluations (MEs)

Datasets	Models	*Flu.*	*Con.*	*Inf.*
Twitter	VPN *vs.* S2STA	56.49	54.92	54.07
	VPN *vs.* DelNet	57.89	60.40	57.50
Weibo	VPN *vs.* S2STA	52.31	52.99	53.54
	VPN *vs.* DelNet	56.56	55.66	54.72

Table 4: Manual evaluations with fluency (*Flu.*), consistency (*Con.*), and informativeness (*Inf.*). The score is the percentage that VPN wins a baseline after removing "tie" pairs. VPN is clearly better than all baselines on the three metrics, and all results are at 99% confidence intervals.

Comparison Settings. Similar to manual evaluations used in Zhou et al. (2018), we conduct a pair-wise comparison between the response generated by VPN and the one for the same input by two typical baselines: S2STA and DelNet. we sample 100 responses from each system, then two curators judge (win, tie and lose) between these two methods.

Comparison Results. The results of manual evaluations are shown in Table 4, where the score is the percentage that VPN wins a baseline after removing "tie" pairs. The Cohen Kappa for inter-annotator statistics is 61.2, 62.1 and 70.8 for fluency, consistency and informativeness, respectively. We can see that our VPN is significantly (sign test, p-value < 0.01) better than all baselines in terms of the three metrics, which further demonstrates that VPN is able to deliver fluent, consistent and informative responses.

4.8 Discussion

Decoders	*Twitter*		*Weibo*	
	BLEU	ROUGE	BLEU	ROUGE
High-Level Dec.	**12.44**	**14.93**	**23.92**	**10.66**
Low-Level Dec.	8.84	8.21	22.50	9.50
Raw-Word Dec.	8.58	7.88	13.02	5.39

Table 5: Performances on each decoder in VPN[8].

The multi-pass decoder in VPN has three decoders. In order to investigate the reasons why the multi-pass decoder works, we will see performances on each decoder in Table 5. We can observe that the high-level decoder obtains the best performances on all metrics, and the low-level decoder outperforms the raw-word decoder. It is intuitive that the high-level decoder performs the best since it has the smallest vocabulary (340), while the raw-word decoder performs the worst because it is equipped with the biggest vocabulary (34000). From the point of performances on each decoder, the effectiveness of multi-pass decoder could be explained from curriculum learning (Bengio et al., 2009). Curriculum learning is a learning strategy in machine learning, where the key idea is to start easier aspects of the targeted task and then gradually increase the complexity. It is difficult for response generation tasks to generate raw words directly. To alleviate this problem, the multi-pass decoder first generates the easier (high-level and low-level) clusters from the small vocabularies, and then generates the raw words from the big vocabulary under the guide of the well-generated clusters. Therefore, the multi-pass decoder obtains significant performances.

5 Related Work

Researches have achieved remarkable improvements on response generation for human-machine conversations. Currently, encoder-decoder framework, especially the Seq2Seq learning (Cho et al., 2014a), is becoming a backbone of data-drive response generation, and it has been widely applied in response generation tasks. For example, Shang et al. (2015a) presented neural recurrent encoder-decoder frameworks for short-text response generation with attention mechanisms (Bahdanau et al., 2015). Li et al. (2016b) introduced persona-based neural response generation to obtain consistent responses for similar inputs to a speaker. Shao et al. (2017) added a self-attention to generate long and diversified responses in Seq2Seq learning.

In this study, we focus on two important problems in response generation: one fixed vocabulary and one-pass decoding. Our work is inspired by following researches to alleviate issues on the fixed vocabulary. Gu et al. (2016) proposed CopyNet, which is able to copy words from the source message. External knowledge bases were also leveraged to extend the vocabulary (Qian et al., 2018; Zhou et al., 2018; Ghazvininejad et al., 2018). Moreover, Xing et al. (2017) incorporated topic words into Seq2Seq frameworks, where topic words are obtained from a pre-trained LDA model (Blei et al., 2003). Wu et al. (2018b) changed the static vocabulary mechanism by a dynamic vocabulary, which jointly learns vocabulary selection and response generation.

We also borrow the idea from studies beyond one-pass decoding. Mou et al. (2016) designed backward and forward sequence generators. Xia et al. (2017) proposed deliberation networks on sequence generation beyond one-pass decoding, where the first-pass decoder generates a raw word sequence, and then the second decoder delivers a refined word sequence based on the raw word sequence. Furthermore, Su et al. (2018) presented hierarchical decoding with linguistic patterns on data-to-text tasks.

However, there has been no unified frameworks to solve the issues of fixed vocabulary and one-pass decoding. Differently, we propose multi-pass encoding and decoding with multi-level vocabularies to deal with the above two problems simultaneously.

6 Conclusion and Future Work

In this study, we tackle the issues of one fixed vocabulary and one-pass decoding in response generation tasks. To this end, we have introduced vocabulary pyramid networks, in which dialogue input and output are represented by multi-level vocabularies and then processed by multi-pass encoding and decoding, where the multi-level vocabularies are obtained from hierarchical clustering of raw words. We conduct experiments on English Twitter and Chinese Weibo datasets. Experiments

[8]In the Discussion Section, all evaluations are based on tokens (IDs) for unifying, so the performances of raw-word decoder on Chinese Weibo dataset are different from the ones (character level) in Table 1.

demonstrate that the proposed method is remarkably better than strong baselines on both automatic and manual evaluations.

In the future, there are some promising explorations in vocabulary pyramid networks. 1) we will further study how to obtain multi-level vocabularies, such as employing other clustering methods and incorporating semantic lexicons like WordNet; 2) we also plan to design deep-pass encoding and decoding for VPN; 3) we will investigate how to apply VPN to other natural language generation tasks such as machine translation and generative text summarization.

Acknowledgments

This work is supported by the Natural Science Foundation of China (No.61533018), the Natural Key R&D Program of China (No.2017YFB1002101), the Natural Science Foundation of China (No.61702512) and the independent research project of National Laboratory of Pattern Recognition. This work is also supported by Alibaba Group through Alibaba Innovative Research (AIR) Program, CCF-DiDi BigData Joint Lab and CCF-Tencent Open Research Fund.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of ICML*, pages 41–48. ACM.

David M. Blei, Andrew Y. Ng, Michael I. Jordan, and John Lafferty. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:2003.

Xinchi Chen, Zhan Shi, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-criteria learning for chinese word segmentation. In *Proceedings of ACL*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014a. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of EMNLP*, pages 1724–1734.

Kyunghyun Cho, B van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014b. On the properties of neural machine translation: Encoder-decoder approaches. In *Proceedings of SSST-8*.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In *Proceedings of AAAI*.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of ACL*, pages 1631–1640.

Shizhu He, Cao Liu, Kang Liu, and Jun Zhao. 2017. Generating natural answers by incorporating copying and retrieving mechanisms in sequence-to-sequence learning. In *Proceedings of ACL*, pages 199–208.

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

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016a. A diversity-promoting objective function for neural conversation models. In *Proceedings of NAACL*, pages 110–119.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016b. A persona-based neural conversation model. In *Proceedings of ACL*, pages 994–1003.

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. 2016c. Deep reinforcement learning for dialogue generation. In *Proceedings of EMNLP*, pages 1192–1202.

Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. 2017. Adversarial learning for neural dialogue generation. In *Proceedings of EMNLP*, pages 2157–2169.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of ACL workshop*, page 10.

Cao Liu, Shizhu He, Kang Liu, and Jun Zhao. 2018. Curriculum learning for natural answer generation. In *Proceedings of IJCAI*, pages 4223–4229.

Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2016. Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. In *Proceedings of COLING*, pages 3349–3358.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for nlg. In *Proceedings of EMNLP*, pages 2241–2252.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318.

Qiao Qian, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Assigning personality/profile to a chatting machine for coherent conversation generation. In *Proceedings of IJCAI*, pages 4279–4285.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *Proceedings of ICLR*.

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

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015a. Neural responding machine for short-text conversation. In *Proceedings of the ACL*, pages 1577–1586.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015b. Neural responding machine for short-text conversation. In *Proceedings of ACL-IJCNLP*, pages 1577–1586. Association for Computational Linguistics.

Yuanlong Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil. 2017. Generating high-quality and informative conversation responses with sequence-to-sequence models. In *Proceedings of EMNLP*, pages 2210–2219.

Shang-Yu Su, Kai-Ling Lo, Yi Ting Yeh, and Yun-Nung Chen. 2018. Natural language generation by hierarchical decoding with linguistic patterns. In *Proceedings of NAACL*, pages 61–66.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of NIPS*, pages 3104–3112.

Alan M Turing. 1950. Computing machinery and intelligence. In *Parsing the Turing Test*, pages 23–65. Springer.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *Proceedings of ICML workshop*.

Joe H. Ward. 1963. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244.

Yu Wu, Wei Wu, Can Xu, and Zhoujun Li. 2018a. Knowledge enhanced hybrid neural network for text matching. In *Proceedings of AAAI*.

Yu Wu, Wei Wu, Dejian Yang, Can Xu, and Zhoujun Li. 2018b. Neural response generation with dynamic vocabularies. In *Proceedings of AAAI*.

Yingce Xia, Fei Tian, Lijun Wu, Jianxin Lin, Tao Qin, Nenghai Yu, and Tie-Yan Liu. 2017. Deliberation networks: Sequence generation beyond one-pass decoding. In *Proceedings of NIPS*, pages 1782–1792.

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic aware neural response generation. In *Proceedings of AAAI*, pages 3351–3357.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In *Proceedings of IJCAI*, pages 4623–4629.