HL-EncDec: A Hybrid-Level Encoder-Decoder for Neural Response Generation

Sixing Wu[*], **Dawei Zhang**[†], **Ying Li** [‡§], **Xing Xie**[†], **Zhonghai Wu**[‡]
[*]School of Software and Microelectronics, Peking University, Beijing, China
[‡]National Research Center of Software Engineering, Peking University, Beijing, China
[†]Microsoft Research Asia, Beijing, China
[§]li.ying@pku.edu.cn

Abstract

Recent years have witnessed a surge of interest on response generation for neural conversation systems. Most existing models are implemented by following the Encoder-Decoder framework and operate sentences of conversations at word-level. The word-level model is suffering from the Unknown Words Issue and the Preference Issue, which seriously impact the quality of generated responses, for example, generated responses may become irrelevant or too general (i.e. safe responses). To address these issues, this paper proposes a hybrid-level Encoder-Decoder model (HL-EncDec), which not only utilizes the word-level features but also character-level features. We conduct several experiments to evaluate HL-EncDec on a Chinese corpus, experimental results show our model significantly outperforms other non-word-level models in automatic metrics and human annotations and is able to generate more informative responses. We also conduct experiments with a small-scale English dataset to show the generalization ability.

1 Introduction

Nowadays, conversation systems have gained a great progress due to the rapid development of big-data and deep learning techniques. A conversation system enables a computer to make human-like conversations with users. Massive conversational datasets can be easily accessed on the Web, which highly promotes both the academia and industry to turn to their attention to develop data-driven conversation systems (Vinyals and Le, 2015; Shang et al., 2015; Chen et al., 2017). A common approach to build a generation-based conversation system is to model it as a response generation task. And a response generation model is often learnt within the Encoder-Decoder framework from large-scale conversational data from the Web.

The Encoder-Decoder is a state-of-the-art framework for SEQ2SEQ (sequence-to-sequence) tasks such as machine translation, abstractive summarization and response generation, etc.. Researchers have proposed several Encoder-Decoder based models, and most of them utilize Recurrent Neural Networks to build an Encoder and an attentional Decoder. Due to the fact that a word is the most basic unit in linguistics, and other larger language elements are built on words, the majority of models operate sentences at word-level. Namely, sentences should be segmented as sequences of words, thus an Encoder-Decoder model could read or generate a sentence with their own vocabularies. Since the computational complexity and the capacity limitation of internal high-speed storage such as GPU VRAM, vocabularies could only record finite words. However, there is an issue that the amount of total words is far more than the volume of vocabularies and the out-of-vocabulary words (denoted as OOVs) may appear in the conversations. Meanwhile, another issue is that words already recorded in the vocabulary have different probabilities to be used in human's dialogues, only a small part of high-frequency words may be learnt well and the remaining low-frequency words may be undertrained. We respectively called the first and

This work is partly supported by National Key R&D Program of China (Grant No. 2017YFB1002002).

Proceedings of the 27th International Conference on Computational Linguistics, pages 845–856
Santa Fe, New Mexico, USA, August 20-26, 2018.

the second issue as Unknown Words Issue and Preference Issue. Intuitively, both issues seriously impact the generation quality and diversity of a response generation model.

Operating sentences at character-level could address the above issues, for the amount of a language's all characters is fixed and small, for example, the alphabet size of English is 128 (ASCII), which allows a vocabulary to record all characters easily. By representing a word as several characters, two issues could be naturally addressed. (Kim et al., 2016) obtained a word embedding by utilizing a convolutional network with character embeddings of that word instead of directly looking up the word embedding matrix of vocabulary. (Costajussa and Fonollosa, 2016) borrowed this idea and proposed a character-level Encoder, but the entire model does not fully work at character-level. (Lee et al., 2016) proposed a fully character-level Encoder-Decoder model, and initially addressed the previous issues. Letting a model work at subword-level is another choice. The BPE (byte-pair-encoding) algorithm proposed by (Sennrich et al., 2016) is able to mine a specific number of most frequent subword units from the corpora and utilizes them to completely represent sentences. (Chung et al., 2016) utilized BPE to build a decoder.

However, these non-word-level models could sharpen the long-term dependencies issue (Hochreiter and Schmidhuber, 1997) and lost much semantic and syntactic information. And those non-word-level approaches are proposed for the machine translation or language modelling instead of response generation task. To the best of our knowledge, there is no existing work that tries to address Unknown Words Issue and Preference Issue for response generation task. In this work, we focus on addressing these two issues for Encoder-Decoder based response generation model by proposing a model named HL-EncDec. HL-EncDec is an improved Encoder-Decoder based model that operates sentences at hybrid-level, which employs characters and words to represent sentences at the same time. In the source side, HL-EncDec is able to utilize a convolutional network to calculate an equivalent word-level embedding vector from characters of that word. Since characters are fully recorded , we could calculate a well-trained equivalent embedding for any word. Moreover, HL-EncDec could also utilize the original word-level embeddings recorded in the matrix by using the sum of equivalent one and original one, this is the main reason why we claim our model works at hybrid-level. In the target side, we also propose a hybrid-level approach to represent a sentence without the increasing of the complexities.

We compare our HL-EncDec with the most widely used standard Encoder-Decoder implementation and other state-of-the-art non-word-level methods using both automatic metrics and human evaluations. The experimental result shows HL-EncDec delivers more high-quality and informative responses compared to baseline models.

2 Background and Motivation

Inspired by SEQ2SEQ approaches for neural machine translation (Sutskever et al., 2014; Bahdanau et al., 2014; Jean et al., 2015), researchers have started extending these techniques to implement generation-based response generation model and already harvested significant improvements. A standard SEQ2SQ model takes a source sentence $X = (x_1, x_2, \ldots, x_T)$ as an input, and generates another sentence $Y = (y_1, y_2, \ldots, y_L)$ as an output. The word-level RNN Encoder-Decoder (denoted as EncDec) is the most famous and applied framework.

In general, an EncDec model has two components: an Encoder first reads a message from user and summarizes this message into several internal representations, and a Decoder utilizes these internal representations to generate a new sentence as a response to reply user's message.

2.1 Encoder

Formally, given a tokenized source sentence $X = (x_1, x_2, \ldots, x_T)$, Encoder reads each word in order and generates hidden states $\mathbf{H} = (\mathbf{h_1}, \mathbf{h_2}, \ldots, \mathbf{h_T})$ as the internal representation of X with a Recurrent Neural Network (RNN). The hidden state $\mathbf{h_t}$ is considered to have already summarized a slice of words $(x_1, \ldots, x_i)$, which is calculated by:

$$\mathbf{h_t} = f(\mathbf{x_t^w}, \mathbf{h_{t-1}}) \tag{1}$$

where function f is always a long-short term memory unit (LSTM) (Hochreiter and Schmidhuber, 1997) or gate recurrent unit (GRU) (Cho et al., 2014), both of them are able to effectively reduce the impact of

long-term dependencies. In this paper, the LSTM is employed as function f. In addition, suppose that x_t is the i-th word of Encoder's vocabulary , thus the embedding vector $\mathbf{x_t^w}$ of word x_t is obtained by an embedding matrix looking up operation:

$$\mathbf{x_t^w} = lookingup(\mathbf{V^e}, x_t) = \mathbf{V^e}[i] \tag{2}$$

where V^e is Encoder's vocabulary and $\mathbf{V^e} \in \mathbb{R}^{|V^e|*d}$ is the corresponding embedding matrix.

Recently, the biRNN (bi-directional RNN) is a common approach to enhance the Encoder (Chung et al., 2014). A biRNN includes a forward RNN and a backward RNN, the forward RNN reads a sentence in its original order while the backward RNN reads it in the reversed order. Thus, the concatenation $\mathbf{h_t^c}$ of the hidden state $\mathbf{h_t^f}$ generated by forward RNN and the hidden state $\mathbf{h_t^b}$ generated by backward RNN is used to replace the single-directional $\mathbf{h_t}$ in the Equation 1.

2.2 Attentional Decoder

Decoder utilizes the hidden states $\mathbf{H}$, and employs another RNN to predict the conditional probability of the next target word $y_{t'}$ being the k-th word of the vocabulary of Decoder V^d at time $t^{'}$:

$$p(y_{t'} = w_k^d | y_{t'-1}, \ldots, y_1, X) = \frac{exp(z(\mathbf{h_t^{'}}, \mathbf{V^d}[k], \mathbf{c_{t'}}))}{\sum_{i=1}^{|V^d|} exp(z(\mathbf{h_t^{'}}, \mathbf{V^d}[i], \mathbf{c_{t'}}))} \tag{3}$$

Where z is a non-linear is function with activation, and $\mathbf{h_{t'}^{'}}$ is hidden state of Decoder's RNN for time $t^{'}$, which also applies a LSTM or GRU to calculate:

$$\mathbf{h_{t'}^{'}} = f(\mathbf{y_{t'-1}}, \mathbf{h_{t'-1}^{'}}, \mathbf{c_{t'}}) \tag{4}$$

The context vector $\mathbf{c_{t'}}$ is computed as a weighted sum of the hidden states $\mathbf{H} = (\mathbf{h_1}, \mathbf{h_2}, \ldots, \mathbf{h_T})$:

$$\mathbf{c_{t'}} = \sum_{i=1}^{T} \alpha_{t',i} \mathbf{h_i} \tag{5}$$

where weight $\alpha_{t',i}$ is given by:

$$\alpha_{t',i} = \frac{exp(e_{t',i})}{\sum_{j=1}^{T} exp(e_{t',j})} \tag{6}$$

$$e_{t',j} = a(\mathbf{h_{t'-1}^{'}}, \mathbf{h_j}) \tag{7}$$

Where a is an alignment model which is used to score how well the word x_j of the input sentence and the next target word $y_t^{'}$ could be related. (Bahdanau et al., 2014) and (Luong et al., 2015) have proposed two different but effective alignment models, respectively. This paper takes the model of (Luong et al., 2015) as function a, which is computed by employing a bilinear term $\mathbf{W_a}$:

$$a(\mathbf{h_{t'-1}^{'}}, \mathbf{h_j}) = (\mathbf{h_{t'-1}^{'}})^{\mathbf{T}} \mathbf{W_a} \mathbf{h_j} \tag{8}$$

2.3 Training

An Encoder-Decoder model can be trained end-to-end by minimizing the negative conditional log-likelihood of target Y given source X, which is defined as:

$$\pounds = -\frac{1}{N} \sum_{i=1}^{N} \sum_{t'=1}^{|Y^i|} \log(p(y_{t'}^i | y_{t'-1}^i, \ldots, y_1^i, X^i) \tag{9}$$

Where N is the total count of the given corpus D , (X^i, Y^i) is the i-th pair of D, and $y_{t'}^i$ is the $t^{'}$-th word of Y^i. is the $t^{'}$-th word of Y^i.

2.4 Challenges

As described above, a word-level EncDec model operates input messages and output responses at word-level. A word itself could not be utilized by a computer unless it has been converted into the internal representation such as a one-hot vector or an embedding vector. Both of Encoder and Decoder maintain their own vocabulary, such that they are able to understand the meaning or predict the probability of a word by looking up the embedding matrix of their vocabularies.

More or less, due to the complexity of model itself, deep neural network models are suffering from the limited computational power of hardware and the scarce capacity of internal high-speed storages such as the GPU VRAM. An EncDec response generation model is surely not an exception, hence the capacity of a vocabulary of the Encoder or Decoder must be controlled to a finite and relatively small number, for example, 40K. Consequently, a lot of words become out-of-vocabulary words (denoted as OOVs), and an EncDec model could not correctly understand or generate it. It is a common approach to enable an EncDec model to work with OOVs that utilizing a special and universal symbol **unk** to replace all OOVs. In other words, all OOVs share a same embedding vector in the source side and they will never be correctly generated in the target side. Intuitively, the more OOVs a corpus contains, the lower performance an EncDec conversation model has. The challenge of the appearance of OOVs is named **Unknown Words Issue**, for convenience.

If we only consider the words already been recorded in the vocabulary (i.e. known words), another challenge will rise to the surface, which is called **Preference Issue**. All known words could be understood and generated by an EncDec model, but that doesn't mean they have the same chance to appear in the sentences of dialogues. Some words are very potential to appear in our dialogues (for example, what, that, I, etc.) and some words are rare to be adopted while it is not an OOV word. This paper calls the first kind of words as high-frequency words (HF words) and the second kind as low-frequency words (LF words). The embedding vector of HF words would be well learnt but LF words could not, for HF words have more appearance to enable them to be updated by commonly used gradient descent optimizing algorithms. Thus, an EncDec model may not correctly operate LF words, which may sharpen the phenomenon that an EncDec conversation model always generates some too safe, too general but very boring responses (e.g., I think so).

The above two issues have seriously impacted the overall performance of EncDec response generation models, even models to solve other tasks such as neural machine translation and neural abstractive text summarization. In this paper, we propose a novel model HL-EncDec to solve these issues, which has significantly improved the generation quality and diversity.

3 Related Work

In linguistics, a word is the most basic unit, and other larger language elements such as phrases and sentences, etc., are built upon words. Due to this fact, it is natural that most previous works regard a word as the smallest unit in their models, i.e. these models operate at word-level.

The word-level EncDec models have achieved great improvements on neural machine translations (Bahdanau et al., 2014; Jean et al., 2015), which inspired researchers to apply this idea to response generation task to build a generation based neural conversation systems(Vinyals and Le, 2015; Shang et al., 2015; Xing et al., 2016; Chen et al., 2017). This technique enables a conversation system to be end-to-end trained from a large-scale corpus of message-response pairs, but researchers found it tends to generate some general and 'safe' responses(Vinyals and Le, 2015). To address this challenge, (Li et al., 2016) introduced a new objective function with MMI (maximum mutual information) to penalize too general responses. (Shao et al., 2017) presented a novel attentional model to generate long and diverse responses. (Wu et al., 2017) enhanced the quality and diversity of generated responses by constructing a dynamic vocabulary.

In this paper, we focus on two issues discussed in Section 2: Unknown Words Issue and Preference Issue. Actually, many researchers have raised their attention to these two issues, and proposed models operate at non-word-level such as character-level and subword-level. (Kim et al., 2016) solved the representation of unknown words and selection preference for language modelling by utilizing characters

to calculate an equivalent word embedding instead of looking up a word embedding matrix. (Costajussa and Fonollosa, 2016) absorbed this idea and proposed a character-level Encoder for machine translation. Recently, inspired by (Kim et al., 2016; Costajussa and Fonollosa, 2016), (Lee et al., 2016) introduced a fully character-level EncDec model. (Sennrich et al., 2016) mined subword units to represent all words by applying BPE algorithm. (Chung et al., 2016) utilized the BPE to build a subword-level Encoder and proposed a character-level Decoder.

While a character-level model or a subword-level model may address the Unknown Words Issue, but it could sharpen the long-term dependencies issue and lost much semantic and syntactic information due to each word must be decomposed to several characters or subunits. The loss may more than gains, and we consider this is why most models operate sentences at word-level.

In addition, previous models are designed for machine translation task which is actually a different task compared with response generation while they are both SEQ2SEQ tasks. This paper proposes a model HL-EncDec, which utilizes the hybrid-level representation technique to improve the generation quality for response generation task.

In the source side ,for a word x, HL-EncDec not only utilizes the word-level embedding vector $\mathbf{x^w}$ obtained by looking up embedding matrix, but also calculates an equivalent word-level embedding vector $\mathbf{x^c}$ of x. To calculate the $\mathbf{x^c}$, HL-EncDec employs its corresponding character sequence and a Convolutional Neural Network based network. The Encoder of HL-EncDec maintains two vocabularies V_w^e and V_c^e, the first vocabulary V_w^e records a finite number of words and their embedding vectors, the second vocabulary V_c^e records all characters of the language used by current system. In general, the total number of all characters of a language is fixed and small, hence recently GPU devices are very easy to load all characters to its RAM. For example, there are about 128 characters in English alphabet (ASCII). Obviously, the concept of out-vocabulary-character has gone with the wind, and each OOV word could be represented as a sequence of fully known characters.

4 HL-EncDec

4.1 Encoder

HL-EncDec employs CharCNN to calculate the equivalent word-level embedding $\mathbf{x^c}$. CharCNN is initially presented for language modeling by (Kim et al., 2016).

Given a word x , CharCNN first represents x as a sequence of characters $C_x = (c_1, \ldots, c_{N_x}, c_{eos})$ where N_x is the character number of x and c_{eos} is a special symbol to indicate the end of characters. Each character c_i could be represented as an embedding vector $\mathbf{c_i} \in \mathbb{R}^{d_c}$ by looking up the embedding matrix $\mathbf{V_c^e} \in \mathbb{R}^{|V_c^e| * d_c}$. Subsequently, a narrow convolution is applied upon the character embeddings $\mathbf{C_x} = (\mathbf{c_1}, \ldots, \mathbf{c_{N_x}}, \mathbf{c_{eos}}) \in \mathbb{R}^{d_c * (N_x+1)}$. Assuming there is a single convolution filter $\mathbf{f} \in \mathbb{R}^{d_c \times w}$ with a variable width w, and the height of $\mathbf{f}$ is fixed to d_c, thus each character vector could be completely operated. With the padding and convolutional operation, the obtained feature map is denoted as $\mathbf{f^m} \in \mathbb{R}^{N_x+1}$, and the i-th element of $\mathbf{f^m}[i]$ is given by:

$$\mathbf{f^m}[i] = ReLU(\sum \mathbf{C_{x[:,i:i+w-1]}} \otimes \mathbf{f}) \tag{10}$$

Where $\otimes$ denotes element-wise matrix multiplication, $\mathbf{C_{x[:,i:i+w-1]}}$ is the i-to-$(i+w-1)$-th column of $\mathbf{C_x}$, which is corresponding to the i-to-$(i+w-1)$-th characters of x. Finally, we take the max-over-time pooling technique:

$$\mathbf{f^p} = \max(\mathbf{f^m}[i]), i \in [1, N_x + 1] \tag{11}$$

Hence, $\mathbf{f^p}$ is regarded as the feature corresponding to the filter $\mathbf{f}$. Assume there are N_f unique filters in total, then $\mathbf{u} \in \mathbb{R}^{N_f}$ is used to denoted the concatenation of corresponding N_f features. Highway network (Hinton et al., 2012) has been shown to bring significant improvements to many NLP tasks, We apply a 4-layer highway network to $\mathbf{u}$, and a single layer highway network is calculated as:

$$\mathbf{u_h^1} = g \odot ReLU(\mathbf{W_h u} + \mathbf{b_h}) + (1 - g) \odot \mathbf{u} \tag{12}$$

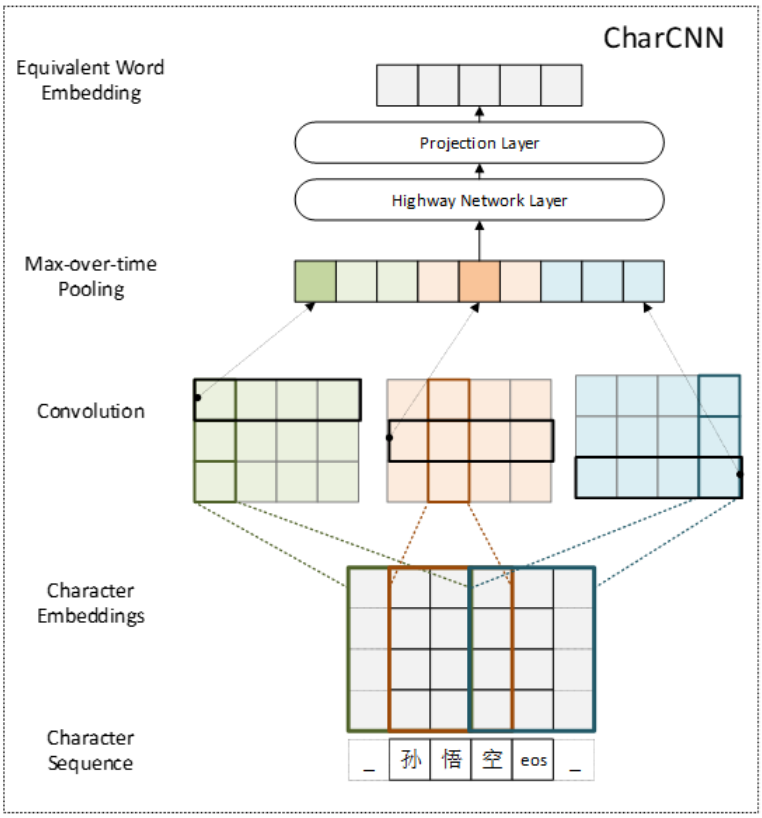

Figure 1: The architecture of CharCNN

where g is called as transform gate, $(1-g)$ is called as carry gate, and $g = sigmod(\mathbf{W_g u} + \mathbf{b_g})$. The equivalent word embedding $\mathbf{x^c}$ is finally worked out by a linear projection:

$$\mathbf{x^c} = \mathbf{u_h^4 W_l} + \mathbf{b_l} \tag{13}$$

Thus, $\mathbf{x^c}$ is the equivalent word embedding calculated by CharCNN, which has the same dimension as $\mathbf{x^w}$ obtained by looking up $\mathbf{V_w^e} \in \mathbb{R}^{|V_w^e| * d_w}$. Then, we calculate the sum of $\mathbf{x^c}$ and $\mathbf{x^w}$:

$$\mathbf{x^h} = \mathbf{x^c} + \mathbf{x^w} \tag{14}$$

Finally, $\mathbf{x^h}$ is the hybrid-level embedding vector for x, and HL-EncDec replaces the word-level $\mathbf{x^w}$ in Equation 2 with $\mathbf{x^h}$. Since $\mathbf{x^c}$ could be generated for any word without the challenge of unknown words and preference, the $\mathbf{x^h}$ could naturally address the Unknown Words Issue and Preference Issue. We also tried to merge $\mathbf{x^w}$ and $\mathbf{x^c}$ with more complex techniques, but this simple summation yield the best result.

4.2 Hybrid-Level Decoder

This paper utilizes a hybrid-level Decoder to avoid the appearance of OOVs and improve the quality and diversity of generated responses. HL-Decoder is able to generate either a word or a character. HL-Decoder has a vocabulary V_d and a corresponding embedding matrix $\mathbf{V_d} \in \mathbb{R}^{|V_d| * d_w}$, which records all characters of the target language, a connector symbol and some HF words. Characters and words are equally recorded in the vocabulary of HL-EncDec, namely HL-Decoder has none explicit operation to judge whether it should generate a character or word, for reducing the complexity of decoding.

While HL-Decoder has another way to segment a sentence into a sequence, HL-Decoder has the same network architecture as the Decoder introduced in Section 2. In the target side of HL-EncDec, a response sentence should be applied a hybrid-level approach to a segment. Assuming there is a tokenized word-level message sequence $Y = (y_1, y_2, \ldots, y_L)$, repeatedly execute the following operation until there is

no OOV word: if y_i is an OOV word, then y_i should be decomposed to several corresponding characters $(c_1, cs, c_2, \ldots, cs, c_{N_y})$, where$(c_1, c_2, \ldots, c_{N_y})$ is the corresponding character sequence of y_i and cs is the connector. Hence, the message sequence could be represented as

$$Y^{'} = (y_1, \ldots, y_{i-1}, c_1, cs, c_2, \ldots, cs, c_{N_y}, y_{i+1}, y_L) \tag{15}$$

Hence, we finally obtain a sequence Y^h without OOVs, and HL-EncDec let the hybrid-level Y^h to replace the word-level Y . Therefore, HL-Decoder is able to generate any word, no matter this word has been recorded in Decoder's vocabulary or not.

5 Experiment

5.1 Model

The widely used word-level model and other famous non-word-level models are selected as baselines:

EncDec: The standard word-level Encoder-Decoder, which has been elaborated in the Section 2. EncDec has a bi-directional LSTM Encoder and a single-directional LSTM Attentional Decoder.

CharEncDec: A character-level EncDec, it has the same network architecture with EncDec, but it fully operates sentences at character-level. Namely, the CharEncDec directly reads or generates a sequence of characters.

BpeEncDec: A subword-level EncDec, and the BPE algorithm is selected to construct the subword units (Sennrich et al., 2016).

CNNEncDec: A fully character-level model proposed by Lee, which employs a character-level convolutional network with max-pooling at Encoder to operate sentence and a character-level Decoder to generate responses (Lee et al., 2016).

For all models, we set the embedding vector dimension as 512, RNN hidden unit size as 256, and batch size as 256. There is an exception in HL-EncDec's source side, the word embedding vector dimension d_w is set to 512 but the character embedding vector dimension d_c is set to 160, for efficiency. The training will be automatically stopped if the perplexity results of dev set are continuously increased in two periods, which is either the end of an epoch or every 10000 global steps. For each model, the beam search with $k = 10$ is applied to infer the responses. All models are implemented by Tensoflow.

5.2 Dataset & Vocabulary

We evaluate HL-EncDec and other models on a Chinese corpus released by Shang, which consists of 4.44 million message-response conversation pairs obtained from Sina Weibo (Shang et al., 2015). Each sentence already been tokenized to words by Stanford Chinese Word Segmenter. After filtering out noisy symbols and duplications, about 3.78 million pairs are finally used. We divide the filtered dataset into 3 sets, there are 20K/20K/3.74M pairs in the test/dev/training sets, respectively.

The vocabulary configurations should be noted here. For character-level models, CharEncDec and CNNEncDec, we keep all characters in vocabularies, and there are 6,690/10,432 characters in the source/target side. For the word-level model EncDec, we individually keep 40,000 most frequent words in the source and target side. For BpeEncDec, we allow BPE algorithm to mine at most (40,000 - 6,454/10,432) subword units in source/target side. For HL-EncDec, we keep 10,432 characters and 29,568 most frequent words in Decoder's vocabulary. In HL-EncDec's source side, we keep 6,690 characters in character vocabulary and keep 38328 words in word vocabulary.

5.3 Metrics

In this paper, we evaluate the performance of models with the following metrics:

BLEU & ROUGE: In previous work, BLEU & ROUGE are already been widely used to evaluate the overall performance of generation quality (Tian et al., 2017; Wu et al., 2017). BLEU is able to configure the max n-gram in the evaluating, hence we configure it from 2 to 4 and respectively denote them as: BLEU-2, BLEU-3 and BLEU-4.

Table 1: Word-level automatic metrics results. A number in bold means the best.

Model	ROUGE	BLEU-2	BLEU-3	BLEU-4	Distinct-1	Distinct-2
EncDec(Dev)	10.75	4.69	3.15	2.44	6.34%	23.48%
EncDec(Test)	10.69	4.60	3.02	2.30	6.28%	23.23%
BpeEncDec(Dev)	**11.45**	5.18	3.40	2.58	6.47%	24.98%
BpeEncDec(Test)	**11.63**	5.15	3.32	2.48	6.37%	24.60%
CNNEncDec(Dev)	9.13	4.00	2.49	1.79	3.85%	16.34%
CNNEncDec(Test)	9.22	3.93	2.39	1.69	3.93%	16.62%
HL-EncDec (Dev)	11.19	**5.28**	**3.55**	**2.74**	**7.08%**	**26.02%**
HL-EncDec (Test)	11.22	**5.26**	**3.53**	**2.74**	**7.03%**	**26.02%**

Table 2: Character-level automatic metrics results. A number in bold means the best.

Model	ROUGE	BLEU-2	BLEU-3	BLEU-4	Distinct-1	Distinct-2
EncDec(Dev)	11.13	6.26	4.42	3.42	1.18%	13.10%
EncDec(Test)	11.09	6.21	4.29	3.24	1.16%	12.87%
CharEncDec(Dev)	11.00	6.18	4.33	3.30	0.86%	12.82%
CharEncDec(Test)	10.83	5.91	4.08	3.07	0.85%	12.69%
BpeEncDec(Dev)	**12.43**	6.68	4.67	3.59	**1.43%**	16.76%
BpeEncDec(Test)	**12.60**	6.69	4.61	3.48	**1.42%**	16.56%
CNNEncDec(Dev)	10.28	5.85	3.83	2.78	0.86%	7.39%
CNNEncDec(Test)	9.22	3.93	2.39	1.69	0.86%	7.39%
HL-EncDec (Dev)	12.37	**6.80**	**4.80**	**3.73**	1.40%	**17.15**%
HL-EncDec (Test)	12.40	**6.83**	**4.77**	**3.68**	1.38%	**17.05**%

DISTINCT-1 & DISTINCT-2: Following (Li et al., 2016; Xing et al., 2016; Wu et al., 2017), we employ the DISTINCT-1/2 to measure how diverse and informative the generated responses are. DISTINCT-1/2 is the ratio of distinct uni-grams/bigrams in generated responses.

Human Evaluation: We employ three native speakers to individually annotate 50 randomly selected groups of responses. Each response may be rated as the following three criteria: +2/Excellent: the response is reasonable, fluent; +1/Acceptable: The response is a little 'safe' or irrelevant or has other small problems; 0/Bad: The response is irrelevant, or ungrammatically or too 'safe'.

5.4 Evaluation Results

Table 1 reports the evaluation results on automatic metrics. The proposed HL-EncDec outperforms baseline models on most metrics. In comparison with the standard EncDec model, HL-EncDec notably outperforms it in all metrics, which demonstrates HL-EncDec is able to generate a more excellent and informative response. Another CNN based model, CNNEncDec, has poor results, we believe the reason behind this is that CNNEncDec could not adapt our Chinese dataset. BPE is one of the state-of-the-art approaches to address the unknown words issue in neural machine translation. Here, BPE keeps this advantage cause BpeEncDec performs better than other baseline models. BpeEncDec only slightly outperforms HL-EncDec in the ROUGE, however, HL-EncDec outperforms BpeEncDec on the other metrics. In the metric BLEU-N, a higher N means it will evaluate the generated responses with more stringent standards. We find that HL-EncDec exceeds BpeEncDec more with the increasement of N, which shows the effectiveness of HL-EncDec from another aspect.

Unlike other languages such as English, etc., a Chinese native speaker always writes or reads a sentence without the explicit segmentation. Considering this fact, we evaluate all models at character-level, namely, we regard each sentence as a character sequence without explicit segmentation. The character-level evaluation results have been reported at Table 2. Clearly, the HL-EncDec still yield the best score on most metrics.

Table 3 reports human evaluation results. HL-EncDec has the highest total score, which means three

Table 3: Human Annotation

Model	EncDec	CharEncDec	BpeEncDec	CNNEncDec	HL-EncDec
0	48.67%	51.33%	32.67%	41.33%	34.00%
1	25.33%	27.33%	30.67%	26.00%	20.00%
2	26.00%	21.33%	36.67%	32.67%	46.00%
Total	38.67	35.00	52.00	45.67	56.00

annotators think our model has the best performance. Similar to the previous results of automatic metrics, although BpeEncDec is behinds HL-EncDec, it still outperforms than other models. EncDec has the lowest score, since EncDec may generate a sentence with unknown words. An unknown word often ruins the whole sentence. In addition, HL-EncDec also has the highest potential to generate a high-quality response, because 46% of responses generated by HL-EncDec are rated as +2/Excellent responses. CNNEncDec beats the EncDec and CharEncDec here, which is different from the automatic metrics results.

Table 4: Case Study

Case	1	2	3
Message	一天两包咖啡已然不够了。	子时已过一小时了，果断睡觉。	晚上吃素，自己动手，啦啦啦。
	It's not enough to drink two packs of coffee a day.	At two in the morning, go to bed	Eat vegetarian at night, DIY， La lala
EncDec	一天一杯，一瓶咖啡。	我还以为是**unk**呢	我也想吃这种蔬菜
	One cup of coffee a day, a bottle of coffee.	I thought it was **unk**	I want to eat this kind of vegetable, too
CharEncDec	一天两包咖啡已然不够了。	子时你过一小时了，果断睡觉。	好可爱啊，好可爱哦
	It's not enough to drink two packs of coffee a day.	At two in the morning, go to bed	It's cute, so cute.
BpeEncDec	一天一天，一天一天	我现在还在睡觉呢。	这就是传说中的胡萝卜?
	Day by day, day by day.	I'm still sleeping now.	This is the legendary carrot?
CNNEncDec	这是什么品种的啊?	时时时刻，果断睡觉	这是什么品种啊？求解
	What kind is this?	Time and time, go to bed	What kind is this?
HL-EncDec	我也想喝一包咖啡。	早睡早起身体健康!	我也想吃。
	I'd like to have a pack of coffee, too.	Early to bed and early to rise makes a man healthy, wealth and wise.	I want to eat, too

We give a qualitative case study in Table 4. By observing, it's easy to find HL-EncDec is able to generate more informative and high-quality responses. In the case 1, sentence generated by EncDec is not fluent, CharEncDec retells the message, BpeEncDec generates a strange response, these three models are unqualified here. In the case 2, EncDec suffers from unknown words, CharEncDec retells message again, CNNEncDec gives an incorrect response. BpeEncDec and HL-EncDec perform well. In the last case, while models except CharEncDec generate acceptable responses, HL-EncDec still gives a most suitable response.

Table 5: Word-level automatic metrics results on Twitter dataset.A number in bold means the best.

Model	ROUGE	BLEU-2	BLEU-3	BLEU-4	Distinct-1	Distinct-2
EncDec	5.84	1.39	0.52	0.23	0.89%	3.19%
BpeEncDec	**6.10**	1.41	0.52	0.23	**1.63%**	**6.06%**
HL-EncDec	5.99	**1.54**	**0.63**	**0.29**	1.34%	4.41%
HL-BPE	5.94	1.25	0.53	0.26	1.53%	5.56%

5.5 Discuss: Generalization

In order to evaluate the generalization ability of HL-EncDec, we take some additional small-scale experiments on an English dataset. The original dataset consists of about 377K message-response pairs from Twitter [1]. We only use 215K pairs with moderate lengths (i.e. 4-20 words). For this dataset, we set the embedding vector dimension as 320, the hidden vector size as 240, and batch size as 64. In HL-EncDec's source side, the character embedding vector dimension d_c is set to 160. The beam search $k = 5$ is applied to infer the responses.

We evaluate four models here, EncDec, BpeEncDec, HL-EncDec, and HL-BPE. HL-BPE is a variant of HL-EncDec, which decomposes an unknown word into a sequence of subword units instead of characters in the target side. An English word generally has more characters than a Chinese word, BPE may help HL-EncDec to reduce the sequence length in the target side. For EncDec, BpeEncDec and the target side of HL-EncDec and HL-BPE, their vocabularies are allowed to record 15,000 items. For the source side of HL-EncDec and HL-BPE, their character vocabularies consist of 617 characters, their word vocabularies consist of 14,383 words.

Table 5 shows the experimental results. EncDec and BpeEncDec have similar BLEU scores, but BpeEncDec has the best ROUGE and Distinct scores. Compared with BpeEncDec, HL-EncDec generates responses with higher qualities (i.e. higher ROUGE+BLEU scores), but less informative(i.e. lower Distinct scores). HL-BPE could be seen as a trade-off between HL-EncDec and BpeEncDec. In brief, these experiments show that HL-EncDec works well with other languages, and HL-EncDec is flexible enough to do some adaptive optimization for a specific language, which is very hard for BpeEncDec.

5.6 Discussion: Why hybrid-level?

Currently, while some previous works could alleviate the Preference Issue for word-level EncDec models in some degree (Weng et al., 2017; Wu et al., 2017), the Unknown Words Issue is still a problem with none effective solution. Employing a vocabulary with fixed words is infeasible since words not only numerous but also being created at any time. A word-level model must take extra actions to alleviate the impact of those unknown OOV words, which may involve more challenges. Character-level approaches could address them, but as what we introduced above and the experimental results, these approaches may decrease the performance.

In the hybrid-level model HL-EncDec, our goal is to keep the advantages of word-level models being reserved as much as possible after we addressed mentioned two issues since a word is a smallest and basic unit in linguistics. Hence, we ingeniously absorb the advantages of characters and propose a hybrid-level Encoder and a hybrid-level decoder.

6 Conclusion and Future work

In this paper, we presented a hybrid-level Encoder-Decoder model, HL-EncDec, for response generation task. This hybrid-level approach absorbs the advantages of both word-level models and character-level models and addressed the Unknown Words Issue and Preference Issue. Experimental results show HL-EncDec could deliver more informative, more relevant responses without the appearance of unknown words. In the future, we will mainly investigate how to improve the techniques in the target side, meanwhile, we will also seek for techniques to enhance our hybrid-level representation technique.

[1] this dataset is released by a third-party github user: https://github.com/marsan-ma/chat_corpus (twitter_en.txt.gz)

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation By Jointly Learning To Align and Translate. In *International Conference on Learning Representations*, pages 1–15, sep.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A Survey on Dialogue Systems: Recent Advances and New Frontiers. *ACM SIGKDD Explorations Newsletter*, 19.

Kyunghyun Cho, Bart Van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. *empirical methods in natural language processing*, pages 1724–1734.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. dec.

Junyoung Chung, Kyunghyun Cho, and Yoshua Bengio. 2016. A Character-Level Decoder without Explicit Segmentation for Neural Machine Translation. mar.

Marta R Costajussa and Jose A R Fonollosa. 2016. Character-based Neural Machine Translation. *meeting of the association for computational linguistics*, pages 357–361.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv: Neural and Evolutionary Computing*.

Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On Using Very Large Target Vocabulary for Neural Machine Translation. *Annual Meeting of the Association for Computational Linguistics*, 000:1–10.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. *national conference on artificial intelligence*, pages 2741–2749.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2016. Fully Character-Level Neural Machine Translation without Explicit Segmentation. In *Annual Meeting of the Association for Computational Linguistics*, pages 1693–1703, mar.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. *north american chapter of the association for computational linguistics*, pages 110–119.

Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. *empirical methods in natural language processing*, pages 1412–1421.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. *meeting of the association for computational linguistics*, pages 1715–1725.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural Responding Machine for Short-Text Conversation. In *Annual Meeting of the Association for Computational Linguistics*, pages 1577–1586.

Louis 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. *eprint arXiv:1701.03185*, jan.

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

Zhiliang Tian, Rui Yan, Lili Mou, Yiping Song, Yansong Feng, and Dongyan Zhao. 2017. How to Make Context More Useful? An Empirical Study on Context-Aware Neural Conversational Models. pages 231–236.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *Computer Science*.

Rongxiang Weng, Shujian Huang, Zaixiang Zheng, Xinyu Dai, and Jiajun Chen. 2017. Neural Machine Translation with Word Predictions. aug.

Yu Wu, Wei Wu, Dejian Yang, Can Xu, Zhoujun Li, and Ming Zhou. 2017. Neural Response Generation with Dynamic Vocabularies.

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Weiying Ma. 2016. Topic Aware Neural Response Generation. *national conference on artificial intelligence*, pages 3351–3357.