Combining Character and Word Information in Neural Machine Translation Using a Multi-Level Attention

Huadong Chen[†], **Shujian Huang**[†]*, **David Chiang**[‡], **Xinyu Dai**[†], **Jiajun Chen**[†]
[†]State Key Laboratory for Novel Software Technology, Nanjing University
{chenhd,huangsj,daixinyu,chenjj}@nlp.nju.edu.cn
[‡]Department of Computer Science and Engineering, University of Notre Dame
dchiang@nd.edu

Abstract

Natural language sentences, being hierarchical, can be represented at different levels of granularity, like words, subwords, or characters. But most neural machine translation systems require the sentence to be represented as a sequence at a single level of granularity. It can be difficult to determine which granularity is better for a particular translation task. In this paper, we improve the model by incorporating multiple levels of granularity. Specifically, we propose (1) an encoder with character attention which augments the (sub)word-level representation with character-level information; (2) a decoder with multiple attentions that enable the representations from different levels of granularity to control the translation cooperatively. Experiments on three translation tasks demonstrate that our proposed models outperform the standard word-based model, the subword-based model and a strong character-based model.

1 Introduction

Neural machine translation (NMT) models (Britz et al., 2017) learn to map from source language sentences to target language sentences via continuous-space intermediate representations. Since word is usually thought of as the basic unit of language communication (Jackendoff, 1992), early NMT systems built these representations starting from the word level (Sutskever et al., 2014; Bahdanau et al., 2015; Cho et al., 2014; Weng et al., 2017). Later systems tried using smaller units such as subwords to address the problem of out-of-vocabulary (OOV) words (Sennrich et al., 2016; Wu et al., 2016).

Although they obtain reasonable results, these word or sub-word methods still have some potential weaknesses. First, the learned representations of (sub)words are based purely on their contexts, but the potentially rich information inside the unit itself is seldom explored. Taking the Chinese word 被打伤 (*bei-da-shang*) as an example, the three characters in this word are a passive voice marker, "hit" and "wound", respectively. The meaning of the whole word, "to be wounded", is fairly compositional. But this compositionality is ignored if the whole word is treated as a single unit.

Secondly, obtaining the word or sub-word boundaries can be non-trivial. For languages like Chinese and Japanese, a word segmentation step is needed, which must usually be trained on labeled data. For languages like English and German, word boundaries are easy to detect, but subword boundaries need to be learned by methods like BPE. In both cases, the segmentation model is trained only in monolingual data, which may result in units that are not suitable for translation.

On the other hand, there have been multiple efforts to build models operating purely at the character level (Ling et al., 2015a; Yang et al., 2016; Lee et al., 2017). But splitting this finely can increase potential ambiguities. For example, the Chinese word 红茶 (*hong-cha*) means "black tea," but the two characters means "red" and "tea," respectively. It shows that modeling the character sequence alone may not be able to fully utilize the information at the word or sub-word level, which may also lead to an inaccurate representation. A further problem is that character sequences are longer, making them more costly to process with a recurrent neural network model (RNN).

While both word-level and character-level information can be helpful for generating better representations, current research which tries to exploit both word-level and character-level information only composed the word-level representation by character embeddings with the word boundary information (Ling et al., 2015b; Costa-jussà and

* Corresponding author.

Proceedings of NAACL-HLT 2018, pages 1284–1293
New Orleans, Louisiana, June 1 - 6, 2018.

Fonollosa, 2016) or replaces the word representation with its inside characters when encountering the out-of-vocabulary words (Luong and Manning, 2016; Wu et al., 2016). In this paper, we propose a novel encoder-decoder model that makes use of both character and word information. More specifically, we augment the standard encoder to attend to individual characters to generate better source word representations (§3.1). We also augment the decoder with a second attention that attends to the source-side characters to generate better translations (§3.2).

To demonstrate the effectiveness of the proposed model, we carry out experiments on three translation tasks: Chinese-English, English-Chinese and English-German. Our experiments show that: (1) the encoder with character attention achieves significant improvements over the standard word-based attention-based NMT system and a strong character-based NMT system; (2) incorporating source character information into the decoder by our multi-scale attention mechanism yields a further improvement, and (3) our modifications also improve a subword-based NMT model. To the best of our knowledge, this is the first work that uses the source-side character information for all the (sub)words in the sentence to enhance a (sub)word-based NMT model in both the encoder and decoder.

2 Neural Machine Translation

Most NMT systems follow the encoder-decoder framework with attention mechanism proposed by Bahdanau et al. (2015). Given a source sentence $\mathbf{x} = x_1 \cdots x_l \cdots x_L$ and a target sentence $\mathbf{y} = y_1 \cdots y_j \cdots y_J$, we aim to directly model the translation probability:

$$P(\mathbf{y} \mid \mathbf{x}; \theta) = \prod_{1}^{J} P(y_j \mid \mathbf{y}_{<j}, \mathbf{x}; \theta),$$

where θ is a set of parameters and $\mathbf{y}_{<j}$ is the sequence of previously generated target words. Here, we briefly describe the underlying framework of the encoder-decoder NMT system.

2.1 Encoder

Following Bahdanau et al. (2015), we use a bidirectional RNN with gated recurrent units (GRUs) (Cho et al., 2014) to encode the source sentence:

$$\begin{aligned} \overrightarrow{h_l} &= \mathrm{GRU}(\overrightarrow{h_{l-1}}, s_l; \overrightarrow{\theta}) \\ \overleftarrow{h_l} &= \mathrm{GRU}(\overleftarrow{h_{l-1}}, s_l; \overleftarrow{\theta}) \end{aligned} \tag{1}$$

where s_l is the l-th source word's embedding, GRU is a gated recurrent unit, $\overrightarrow{\theta}$ and $\overleftarrow{\theta}$ are the parameters of forward and backward GRU, respectively; see Cho et al. (2014) for a definition.

The *annotation* of each source word x_l is obtained by concatenating the forward and backward hidden states:

$$\overleftrightarrow{h_l} = \begin{bmatrix} \overrightarrow{h_l} \\ \overleftarrow{h_l} \end{bmatrix}.$$

The whole sequence of these annotations is used by the decoder.

2.2 Decoder

The decoder is a forward RNN with GRUs predicting the translation $\mathbf{y}$ word by word. The probability of generating the j-th word y_j is:

$$P(y_j \mid \mathbf{y}_{<j}, \mathbf{x}; \theta) = \mathrm{softmax}(\begin{bmatrix} t_{j-1} \\ d_j \\ c_j \end{bmatrix})$$

where t_{j-1} is the word embedding of the $(j-1)$-th target word, d_j is the decoder's hidden state of time j, and c_j is the *context vector* at time j. The state d_j is computed as

$$d_j = \mathrm{GRU}\left(d_{j-1}, \begin{bmatrix} t_{j-1} \\ c_j \end{bmatrix}; \theta_d\right).$$

The attention mechanism computes the context vector c_i as a weighted sum of the source annotations,

$$c_j = \sum_{i=1}^{I} \alpha_{j,l} \overleftrightarrow{h_l} \tag{2}$$

where the attention weight α_{ji} is

$$\alpha_{ji} = \frac{\exp(e_{ji})}{\sum_{i'=1}^{I} \exp(e_{ji'})} \tag{3}$$

and

$$e_{jl} = v_a^{\mathrm{T}} \tanh(W_a d_{j-1} + U_a \overleftrightarrow{h_l}) \tag{4}$$

where v_a, W_a and U_a are the weight matrices of the attention model, and e_{jl} is an attention model that scores how well d_{j-1} and $\overleftrightarrow{h_l}$ match.

With this strategy, the decoder can attend to the source annotations that are most relevant at a given time.

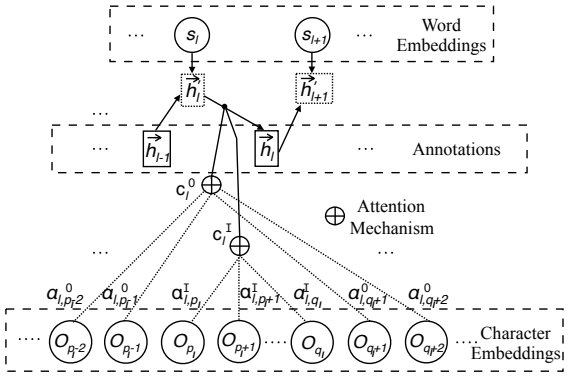

Figure 1: Forward encoder with character attention at time step l. The encoder alternates between reading word embeddings and character context vectors. ${c_l}^I$ and ${c_l}^O$ denotes the inside and outside character-level context vectors of the l-th word, respectively.

3 Character Enhanced Neural Machine Translation

In this section, we present models which make use of both character-level and word-level information in the encoder-decoder framework.

3.1 Encoder with Character Attention

The encoder maps the source sentence to a sequence of representations, which is then used by the attention mechanism. The standard encoder operates purely on (sub)words or characters. However, we want to encode both, since both levels can be linguistically significant (Xiong et al., 2017).

To incorporate multiple levels of granularity, we extend the encoder with two character-level attentions. For each source word, the characters of the whole sentence can be divided into two parts, those inside the word and those outside the word. The inside characters contain information about the internal structure of the word. The outside characters may provide information about patterns that cross word boundaries. In order to distinguish the influence of the two, we use two separate attentions, one for inside characters and one for outside characters.

Note that we compute attention directly from the character embedding sequence instead of using an additional RNN layer. This helps to avoid the vanishing gradient problem that would arise from increasing the sequence length, and also keeps the computation cost at a low level.

Figure 1 illustrates the forward encoder with character attentions. We write the character embeddings as $\mathbf{o} = o_1 \cdots o_k \cdots o_K$. Let p_l and q_l be the starting and ending character position, respectively, of word x_l. Then $o_{p_l} \cdots o_{q_l}$ are the inside characters of word x_l; $o_1 \cdots o_{p_l-1}$ and $o_{q_l+1} \cdots o_K$ are the outside characters of word x_l.

The encoder is an RNN that alternates between reading (sub)word embeddings and character-level information. At each time step, we first read the word embedding:

$$\overrightarrow{h_l}' = \mathrm{GRU}(\overrightarrow{h_{l-1}}, s_l; \overrightarrow{\theta'}) \tag{5}$$

Then we use the attention mechanisms to compute character context vectors for the inside characters:

$$c_l^I = \sum_{m=p_l}^{q_l} \alpha_{lm}^I o_m$$

$$\alpha_{lm}^I = \frac{\exp(e_{lm})}{\sum_{m'=p_l}^{q_l} \exp(e_{lm'})}$$

$$e_{lm} = v^I \cdot \tanh(W^I \overrightarrow{h_l}' + U^I o_m).$$

The outside character context vector ${c_l}^O$ is calculated in a similar way, using a different set of parameters, i.e. W^O, U^O, v^O instead of W^I, U^I, v^I.

The inside and outside character context vectors are combined by a feed-forward layer and fed into the encoder RNN, forming the character-enhanced word representation $\overrightarrow{h_l}$:

$$c_l^C = \tanh(W^I c_l^I + W^O {c_l}^O)$$

$$\overrightarrow{h_l} = \mathrm{GRU}(\overrightarrow{h_l}', c_l^C; \overrightarrow{\theta})$$

Note that this GRU does not share parameters with the GRU in (5).

The backward hidden states are calculated in a similar manner.

3.2 Decoder with Multi-Scale Attention

In order to fully exploit the character-level information, we also make extensions to the decoder, so that the character-level information can be taken into account while generating the translation.

We propose a multi-scale attention mechanism to get the relative information of current decoding step from both word-level and character-level representations. This attention mechanism is build from the high-level to the low-level representation, in order to enhance high-level representation with fine-grained internal structure and context. The multi-scale attention mechanism is built (as shown in Figure 2) from word-level to character-level.

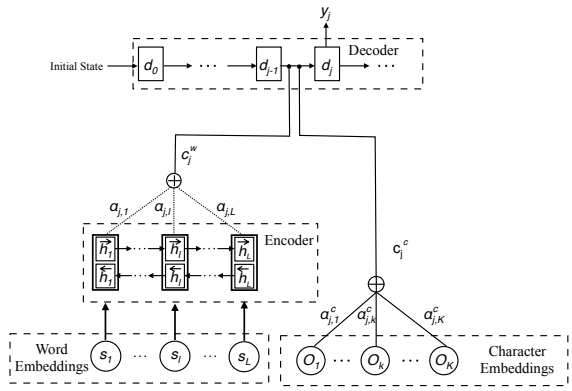

Figure 2: Illustration of the decoder with our multi-scale attention mechanism.

First, we get the word-level information. The context vector c_j^w is calculated following the standard attention model (Eq. 2–4). And the hidden state $\tilde{d}_j$ is updated.

$$\tilde{d}_j = \text{GRU}\left(d_{j-1}, \begin{bmatrix} t_{j-1} \\ c_j^w \end{bmatrix}; \tilde{\theta}_d\right), \tag{6}$$

Then we attend to the character-level representation, which provides more information about the word's internal structure. The context vector c_j^c is calculated based on the updated hidden state above,

$$\begin{aligned} c_j^c &= \sum_{k=1}^{K} \alpha_{jk}^c o_k \\ \alpha_{jk}^c &= \frac{\exp(e_{jk})}{\sum_{k'=1}^{K} \exp(e_{jk'})} \\ e_{j,k} &= v^c \cdot \tanh(W^c \tilde{d}_j + U^c o_k). \end{aligned}$$

Finally, the word-level context vector $c_j^w$and character-level context vector c_j^c are concatenated:

$$c_j = \begin{bmatrix} c_j^w \\ c_j^c \end{bmatrix}.$$

And the final context vector c_j is used to help predict the next target word.

$$P(y_j \mid \mathbf{y}_{<j}, \mathbf{x}; \theta) = \text{softmax}(\begin{bmatrix} t_{j-1} \\ d_j \\ c_j \end{bmatrix})$$

where d_j is

$$d_j = \text{GRU}(\tilde{d}_j, c_j^c; \theta_d),$$

With this mechanism, both the (sub)word-level and character-level representations could be used to predict the next translation, which helps to ensure a more robust and reasonable choice. It may also help to alleviate the under-translation problem because the character information could be a complement to the word.

4 Experiments

We conduct experiments on three translation tasks: Chinese-English (Zh-En), English-Chinese (En-Zh) and English-German (En-De). We write Zh↔En to refer to the Zh-En and En-Zh tasks together.

4.1 Datasets

For Zh↔En, the parallel training data consists of 1.6M sentence pairs extracted from LDC corpora, with 46.6M Chinese words and 52.5M English words, respectively.[1] We use the NIST MT02 evaluation data as development data, and MT03, MT04, MT05, and MT06 as test data. The Chinese side of the corpora is word segmented using ICTCLAS.[2] The English side of the corpora is lowercased and tokenized.

For En-De, we conduct our experiments on the WMT17 corpus. We use the pre-processed parallel training data for the shared news translation task provided by the task organizers.[3] The dataset consitst of 5.6M sentence pairs. We use `newstest2016` as the development set and evaluate the models on `newstest2017`.

4.2 Baselines

We compare our proposed models with several types of NMT systems:

- **NMT:** the standard attentional NMT model with words as its input (Bahdanau et al., 2015).
- **RNN-Char:** the standard attentional NMT model with characters as its input.
- **CNN-Char:** a character-based NMT model, which implements the convolutional neural network (CNN) based encoder (Costa-jussà and Fonollosa, 2016).
- **Hybrid:** the mixed word/character model proposed by Wu et al. (2016).

[1] LDC2002E18, LDC2003E14, the Hansards portion of LDC2004T08, and LDC2005T06.

[2] `http://ictclas.nlpir.org`

[3] `http://data.statmt.org/wmt17/translation-task/preprocessed/de-en/`

System	MT02	MT03	MT04	MT05	MT06	Mean	Δ
NMT	33.76	31.88	33.15	30.55	27.47	30.76	
Word-att	34.28	32.26	33.82	31.02	27.93	31.26	+0.50
Char-att	34.85	33.71	34.91	32.08	28.66	32.34	+1.58

Table 1: Performance of the encoder with character attention and the encoder with word attention. Char-att and Word-att denotes the encoder with character attention and the encoder with word attention, respectively.

- **BPE:** a subword level NMT model, which processes the source side sentence by Byte Pair Encoding (BPE) (Sennrich et al., 2016).

We used the dl4mt implementation of the attentional model,[4] reimplementing the above models.

4.3 Details

Training For Zh↔En, we filter out the sentence pairs whose source or target side contain more than 50 words. We use a shortlist of the 30,000 most frequent words in each language to train our models, covering approximately 98.2% and 99.5% of the Chinese and English tokens, respectively. The word embedding dimension is 512. The hidden layer sizes of both forward and backward sequential encoder are 1024. For fair comparison, we also set the character embedding size to 512, except for the CNN-Char system. For CNN-Char, we follow the standard setting of the original paper (Costa-jussà and Fonollosa, 2016).

For En-De, we build the baseline system using joint BPE segmentation (Sennrich et al., 2017). The number of joint BPE operations is 90,000. We use the total BPE vocabulary for each side.

We use Adadelta (Zeiler, 2012) for optimization with a mini-batch size of 32 for Zh↔En and 50 for En-De.

Decoding and evaluation We use beam search with length-normalization to approximately find the most likely translation. We set beam width to 5 for Zh↔En and 12 for En-De. The translations are evaluated by BLEU (Papineni et al., 2002). We use the `multi-bleu` script for Zh↔En,[5] and the `multi-bleu-detok` script for En-De.[6]

[4] https://github.com/nyu-dl/dl4mt-tutorial

[5] https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl

[6] https://github.com/EdinburghNLP/nematus/blob/master/data/multi-bleu-detok.perl

4.4 Results: Encoder with character attention

This set of experiments evaluates the effectiveness of our proposed character enhanced encoder. In Table 1, we first compare the encoder with character attention (Char-att) with the baseline word-based model. The result shows that our extension of the encoder can obtain significantly better performance (+1.58 BLEU).

Then, in order to investigate whether the improvement comes from the extra parameters in the character layer, we compare our model to a word embedding enhanced encoder. When the word embedding enhanced encoder encodes a word, it attends to the word's embedding and other word embedding in the sentence instead of attending to the word's inside and outside character embeddings. The results show that the word embedding enhanced encoder (Word-att) only gets a 0.5 BLEU improvement than the baseline, while our model is significantly better (+1.58 BLEU). This shows that the benefit comes from the augmented character-level information which help the word-based encoder to learn a better source-side representation.

Finally, we compare our character enhanced model with several types of systems including a strong character-based model proposed by Costa-jussà and Fonollosa (2016) and a mixed word/character model proposed by Wu et al. (2016). In Table 2, rows 2 and 2′ confirm the finding of Yang et al. (2016) that the traditional RNN model performs less well when the input is a sequence of characters. Rows 4 and 4′ indicate that Wu et al. (2016)'s scheme to combine of words and characters is effective for machine translation. Our model (row 5) outperforms other models on the Zh-En task, but only outperforms the word-based model on En-Zh. The results may suggest that the CNN and RNN methods is also strong in building the source representation.

Task	#	System	MT02	MT03	MT04	MT05	MT06	Mean	Δ
Zh-En	1	NMT	33.76	31.88	33.15	30.55	27.47	30.76	
	2	RNN-Char	32.22	31.05	31.41	28.85	25.99	29.32	−1.44
	3	CNN-Char	33.69	32.06	33.10	30.40	27.67	30.81	+0.05
	4	Hybrid	34.33	33.10	33.41	30.96	28.00	31.37	+0.60
	5	Char-att	34.85	33.71	34.91	32.08	28.66	32.34	+1.58
	6	Multi-att	34.61	33.26	34.42	31.06	28.24	31.75	+0.98
	7	5+6	35.42	33.9	35.23	32.62	29.36	32.68	+2.02
En-Zh	1′	NMT	31.58	22.20	23.47	22.50	21.47	22.41	
	2′	RNN-Char	28.78	21.03	21.70	19.81	20.98	20.88	−1.53
	3′	CNN-Char	31.36	23.60	24.71	22.75	23.05	23.53	+1.12
	4′	Hybrid	31.31	24.45	24.65	23.10	23.62	23.96	+1.55
	5′	Char-att	30.93	23.63	24.42	21.92	23.6	23.39	+0.98
	6′	Multi-att	30.17	22.09	24.09	22.29	23.8	23.07	+0.66
	7′	5′+6′	32.91	25.02	25.69	24.03	25.20	24.99	+2.58

Table 2: Performance of different systems on the Chinese-English and English-Chinese translation tasks. Our encoder with character attention (Char-att) improves over all other models on Zh-En and over the word-based baseline on En-Zh. Adding our decoder with multi-scale attention (Multi-att) outperforms all other models.

Task	#	System	MT02	MT03	MT04	MT05	MT06	Mean	Δ
Zh-En	8	BPE	34.66	33.65	34.69	30.80	27.66	31.70	+0.94
	9	Char-att	35.20	34.93	35.39	31.62	28.56	32.63	+1.86
	10	9+Multi-att	36.68	35.39	35.93	32.08	29.74	33.29	+2.52
En-Zh	8′	BPE	30.17	22.09	24.09	22.29	23.80	23.07	+0.66
	9′	Char-att	30.95	23.07	25.19	22.74	24.27	23.82	+1.41
	10′	9′+Multi-att	32.36	24.91	25.79	23.42	24.88	24.75	+2.34

Table 3: Comparison of our models on top of the BPE-based NMT model and the original BPE-based model on the Chinese-English and English-Chinese translation tasks. Our models improve over the BPE baselines.

4.5 Results: Multi-scale attention

Rows 6 and 6′ in Table 2 verify that our multi-scale attention mechanism can obtain better results than baseline systems. Rows 7 and 7′ in Table 2 show that our proposed multi-scale attention mechanism further improves the performance of our encoder with character attention, yielding a significant improvement over the standard word-based model on both Zh-En (+2.02 vs. row 1) task and En-Zh translation task (+2.58 vs. row 1′).

Compared to the CNN-Char model, our model still gets +1.97 and +1.46 BLEU improvement on Zh-En and En-Zh, respectively. Compared to the mixed word/character model proposed by (Wu et al., 2016), we find that our best model gives a better result, demonstrating the benefits of exploiting the character level information during decoding.

System	Dev	Test	Δ
BPE	28.41	23.05	
Char-att	29.80	23.87	+0.82
+Multi-att	30.52	24.48	+1.43

Table 4: Case-sensitive BLEU on the English-German translation tasks. Our systems improve over a baseline BPE system.

4.6 Results: Subword-based models

Currently, subword-level NMT models are widely used for achieving open-vocabulary translation. Sennrich et al. (2016) introduced a subword-level NMT model using subword-level segmentation based on the byte pair encoding (BPE) algorithm. In this section, we investigate the effectiveness of our character enhanced model on top of the BPE model. Table 3 shows the results on the Zh-En task

(a) OOV words

Source	互联网 业务 仍 将 是 中国 **通信业** 增长 速度 最 快 的 业务 [...]
Reference	internet will remain the business with the fastest growth in china 's *telecommunication industry* [...]
NMT	internet business will remain the fastest growing business in china . [...]
Hybrid	the internet will remain the fastest of china 's *communications* growth speed [...]
Ours	internet business will continue to be the fastest growing business of china 's *telecommunications industry* [...]

(b) Frequent words

Source	[...] 不 是 目前 在 巴勒斯坦 **被占领土** 所 发生 的 事件 的 根源 [...]
Reference	[...] not the source of what happened on the palestinian *territory occupied* by israel [...]
NMT	[...] such actions were not the source of the incidents of the current palestinian *occupation* [...]
CNN-Char	[...] not the only source of events that took place in the palestinian *occupation* [...]
Ours	[...] not the root of the current incidents that took place in the palestinian *occupied territories* [...]
Source	[...] 将 **东西方 冷战** 的 象征 柏林 墙 的 三 块 墙体 赠送 到 了 联合国
Reference	[...] presented the united nations with three pieces of the berlin wall , a symbol of *the cold war between the east and the west* .
NMT	[...] sent the three pieces of UNK UNK to the un to the un
CNN-Char	[...] sent three pieces of UNK to the united nations, which was *the cold war in eastern china* .
Ours	[...] presented the un on the 4th of the three wall UNK of *the eastern and western cold war* .

Table 5: Sample translations. For each example, we show the source, the reference and the translation from our best model. "Ours" means our model with both Char-att and Multi-att.

and En-Zh translation task. Rows 8 and 8′ confirm that BPE slightly improves the performance of the word-based model. But both our character enhanced encoder and the multi-scale attention yield better results. Our best model leads to improvements of up to 1.58 BLEU and 1.68 BLEU on the Zh-En task and En-Zh translation task, respectively.

We also conduct experiments on the En-De translation task (as shown in Table 4). The result is consistent with Zh-En task and En-Zh translation tasks. Our best model obtains 1.43 BLEU improvement over the BPE model.

System	**with OOV**	**Δ**	**no OOV**	**Δ**
NMT	28.47		38.21	
Hybrid	29.83	+1.36	37.79	−0.43
Ours	30.80	+2.33	39.39	+1.18

Table 6: Translation performance on source sentences with and without OOV words. "Ours" means our model with both Char-att and Multi-att.

4.7 Analysis

We have argued that the character information is important not only for OOV words but also frequent words. To test this claim, we divided the MT03 test set into two parts according to whether

the sentence contains OOV words, and evaluated several systems on the two parts. Table 6 lists the results. Although the hybrid model achieves a better result on the sentences which contain OOV words, it actually gives a worse result on the sentences without OOV words. By contrast, our model yields the best results on both parts of the data. This shows that frequent words also benefit from fine-grained character-level information.

Table 5 shows three translation examples. Table 5(a) shows the translation of an OOV word 通信业 (*tong-xin-ye*, telecommunication industry). The baseline NMT system can't translate the whole word because it is not in the word vocabulary. The hybrid model translates the word to "communication," which is a valid translation of the first two characters 通信. This mistranslation also appears to affect other parts of the sentence adversely. Our model translates the OOV word correctly.

Table 5(b) shows two translation samples involving frequent words. For the compound word 被占领土 (*beizhanlingtu*, occupied territory), the baseline NMT system only partly translates the word as "occupation" and ignores the main part 领土 (*lingtu*, territory). The CNN-Char model, which builds up the word-level representation from characters, also cannot capture 领土 (*lingtu*). However, our model correctly translates the word as "occupied territories." (The phrase "by Israel" in the reference was inserted by the translator.) The word 东西方 (*dongxifang*, east and west) and 冷战 (*lengzhan*, cold war) are deleted by the baseline model, and even the CNN-Char model translates 东西方 (*dongxifang*) incorrectly. By contrast, our model can make use of both words and characters to translate the word 东西方 (*dongxifang*) reasonably well as "eastern and western."

5 Related Work

Many recent studies have focused on using character-level information in neural machine translation systems. These efforts could be roughly divided into the following two categories.

The first line of research attempted to build neural machine translation models purely on characters without explicit segmentation. Lee et al. (2017) proposed to directly learn the segmentation from characters by using convolution and pooling layers. Yang et al. (2016) composed the high-level representation by the character embedding and its surrounding character-level context with a bidirectional and concatenated row convolution network. Different from their models, our model aims to use characters to enhance words representation instead of depending on characters solely; our model is also much simpler.

The other line of research attempted to combine character-level information with word-level information in neural machine translation models, which is more similar with our work. Ling et al. (2015a) employed a bidirectional LSTM to compose character embeddings to form the word-level information with the help of word boundary information. Costa-jussà and Fonollosa (2016) replaced the word-lookup table with a convolutional network followed by a highway network (Srivastava et al., 2015), which learned the word-level representation by its constituent characters. Zhao and Zhang (2016) designed a *decimator* for their encoder, which effectively uses a RNN to compute a word representation from the characters of the word. These approaches only consider word boundary information and ignore the word-level meaning information itself. In contrast, our model can make use of both character-level and word-level information.

Luong and Manning (2016) proposed a hybrid scheme that consults character-level information whenever the model encounters an OOV word. Wu et al. (2016) converted the OOV words in the word-based model into the sequence of its constituent characters. These methods only focus on dealing with OOV words by augmenting the character-level information. In our work, we augment the character information to all the words.

6 Conclusion

In this paper, we have investigated the potential of using character-level information in word-based and subword-based NMT models by proposing a novel character-aware encoder-decoder framework. First, we extended the encoder with a character attention mechanism for learning better source-side representations. Then, we incorporated information about source-side characters into the decoder with a multi-scale attention, so that the character-level information can cooperate with the word-level information to better control the translation. The experiments have demonstrated the effectiveness of our models. Our analysis showed that both OOV words and frequent

words benefit from the character-level information.

Our current work only uses the character-level information in the source-side. For future work, it will be interesting to make use of finer-grained information on the target side as well.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments. This work is supported by the National Science Foundation of China (No. 61772261 and 61672277) and the Jiangsu Provincial Research Foundation for Basic Research (No. BK20170074). Part of Huadong Chen's contribution was made while visiting University of Notre Dame. His visit was supported by the joint PhD program of China Scholarship Council.

References

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

Denny Britz, Quoc Le, and Reid Pryzant. 2017. Effective domain mixing for neural machine translation.

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. In *Proc. EMNLP*, pages 1724–1734.

Marta R. Costa-jussà and José A. R. Fonollosa. 2016. Character-based neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 357–361, Berlin, Germany. Association for Computational Linguistics.

Ray S. Jackendoff. 1992. *Semantic Structures*. MIT Press.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378.

Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W. Black. 2015a. Character-based neural machine translation. *CoRR*, abs/1511.04586.

Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W Black. 2015b. Character-based neural machine translation. *arXiv preprint arXiv:1511.04586*.

Minh-Thang Luong and D. Christopher Manning. 2016. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1054–1063. Association for Computational Linguistics.

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

Rico Sennrich, Alexandra Birch, Anna Currey, Ulrich Germann, Barry Haddow, Kenneth Heafield, Antonio Valerio Miceli Barone, and Philip Williams. 2017. The university of edinburgh's neural mt systems for wmt17.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pages 2377–2385, Cambridge, MA, USA. MIT Press.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27*, pages 3104–3112.

Rongxiang Weng, Shujian Huang, Zaixiang Zheng, XIN-YU DAI, and Jiajun CHEN. 2017. Neural machine translation with word predictions. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 136–145, Copenhagen, Denmark. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

H. Xiong, Z. He, X. Hu, and H. Wu. 2017. Multi-channel Encoder for Neural Machine Translation. arXiv:1712.02109.

Zhen Yang, Wei Chen, Feng Wang, and Bo Xu. 2016. A character-aware encoder for neural machine translation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3063–3070, Osaka, Japan. The COLING 2016 Organizing Committee.

Matthew D. Zeiler. 2012. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701.

Shenjian Zhao and Zhihua Zhang. 2016. An efficient character-level neural machine translation. *CoRR*, abs/1608.04738.