Is Word Segmentation Necessary for Deep Learning of Chinese Representations?

Yuxian Meng*[♠], Xiaoya Li*[♠], Xiaofei Sun[♠], Qinghong Han[♠]
Arianna Yuan[♠,♡], and Jiwei Li[♠,♣]

[♣] School of Information, Renmin University of China
[♡] Computer Science Department, Stanford University
[♠] Shannon.AI
{ yuxian_meng, xiaoya_li, xiaofei_sun, qinghong_han , jiwei_li}@shannonai.com
xfyuan@stanford.edu

Abstract

Segmenting a chunk of text into words is usually the first step of processing Chinese text, but its necessity has rarely been explored.

In this paper, we ask the fundamental question of whether Chinese word segmentation (CWS) is necessary for deep learning-based Chinese Natural Language Processing. We benchmark neural word-based models which rely on word segmentation against neural char-based models which do not involve word segmentation in four end-to-end NLP benchmark tasks: language modeling, machine translation, sentence matching/paraphrase and text classification. Through direct comparisons between these two types of models, we find that char-based models consistently outperform word-based models.

Based on these observations, we conduct comprehensive experiments to study why word-based models underperform char-based models in these deep learning-based NLP tasks. We show that it is because word-based models are more vulnerable to data sparsity and the presence of out-of-vocabulary (OOV) words, and thus more prone to overfitting. We hope this paper could encourage researchers in the community to rethink the necessity of word segmentation in deep learning-based Chinese Natural Language Processing. [1]

1 Introduction

There is a key difference between English (or more broadly, languages that use some form of the Latin alphabet) and Chinese (or other languages that do not have obvious word delimiters such as Korean and Japanese) : words in English can be easily recognized since the space token is a good approximation of a word divider, whereas no word divider is present between words in written Chinese sentences. This gives rise to the task of Chinese Word Segmentation (CWS) (Zhang et al., 2003; Peng et al., 2004; Huang and Zhao, 2007; Zhao et al., 2006; Zheng et al., 2013; Zhou et al., 2017; Yang et al., 2017, 2018). In the context of deep learning, the segmented words are usually treated as the basic units for operations (we call these models the word-based models for the rest of this paper). Each segmented word is associated with a fixed-length vector representation, which will be processed by deep learning models in the same way as how English words are processed. Word-based models come with a few fundamental disadvantages, as will be discussed below.

Firstly, word data sparsity inevitably leads to overfitting and the ubiquity of OOV words limits the model's learning capacity. Particularly, Zipf's law applies to most languages including Chinese. Frequencies of many Chinese words are extremely small, making the model impossible to fully learn their semantics. Let us take the widely used Chinese Treebank dataset (CTB) as an example (Xia, 2000). Using Jieba,[2] the most widely-used open-sourced Chinese word segmentation system, to segment the CTB, we end up with a dataset consisting of 615,194 words with 50,266 distinct words. Among the 50,266 distinct words, 24,458 words appear only once, amounting to 48.7% of the total vocabulary, yet they only take up 4.0% of the entire corpus. If we increase the frequency bar to 4, we get 38,889 words appearing less or equal to 4 times, which contribute to 77.4% of the total vocabulary but only 10.1% of the entire corpus. Statistics are given in Table 1. This shows that the word-based data is very sparse. The data sparsity issue is likely to induce overfitting, since more words means a larger number of parameters. In addition, since it

[1] Yuxian Meng and Xiaoya Li contributed equally to this paper.

[2] https://github.com/fxsjy/jieba

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3242–3252
Florence, Italy, July 28 - August 2, 2019.

bar	# distinct	prop of vocab	prop of corpus
∞	50,266	100%	100%
4	38,889	77.4%	10.1%
1	24,458	48.7%	4.0%

Table 1: Word statistics of Chinese TreeBank.

Corpora	Yao	Ming	reaches	the final	
CTB	姚明		进入	总决赛	
PKU	姚	明	进入	总	决赛

Table 2: CTB and PKU have different segmentation criteria (Chen et al., 2017c).

is unrealistic to maintain a huge word-vector table, many words are treated as OOVs, which may further constrain the model's learning capability.

Secondly, the state-of-the-art word segmentation performance is far from perfect, the errors of which would bias downstream NLP tasks. Particularly, CWS is a relatively hard and complicated task, primarily because word boundary of Chinese words is usually quite vague. As discussed in Chen et al. (2017c), different linguistic perspectives have different criteria for CWS (Chen et al., 2017c). As shown in Table 1, in the two most widely adopted CWS datasets PKU (Yu et al., 2001) and CTB (Xia, 2000), the same sentence is segmented differently.

Thirdly, if we ask the fundamental problem of how much benefit word segmentation may provide, it is all about how much additional semantic information is present in a labeled CWS dataset. After all, the fundamental difference between word-based models and char-based models is whether teaching signals from the CWS labeled dataset are utilized. Unfortunately, the answer to this question remains unclear. For example. in machine translation we usually have millions of training examples. The labeled CWS dataset is relatively small (68k sentences for CTB and 21k for PKU), and the domain is relatively narrow. It is not clear that CWS dataset is sure to introduce a performance boost.

Before neural network models became popular, there were discussions on whether CWS is necessary and how much improvement it can bring about. In information retrieval(IR), Foo and Li (2004) discussed CWS's effect on IR systems and revealed that segmentation approach has an effect on IR effectiveness as long as the SAME segmentation method is used for query and document, and that CWS does not always work better than models without segmentation. In cases where CWS does lead to better performance, the gap between word-based models and char-based models can be closed if bigrams of characters are used in char-based models. In the phrase-based machine translation, Xu et al. (2004) reported that CWS only showed non-significant improvements over models without word segmentation. Zhao et al. (2013) found that segmentation itself does not guarantee better MT performance and it is not key to MT improvement. For text classification, Liu et al. (2007) compared a naïve character bigram model with word-based models, and concluded that CWS is not necessary for text classification. Outside the literature of computational linguistics, there have been discussions in the field of cognitive science. Based on eye movement data, Tsai and McConkie (2003) found that fixations of Chinese readers do not land more frequently on the centers of Chinese words, suggesting that characters, rather than words, should be the basic units of Chinese reading comprehension. Consistent with this view, Bai et al. (2008) found that Chinese readers read unspaced text as fast as word spaced text.

In this paper, we ask the fundamental question of whether word segmentation is necessary for deep learning-based Chinese natural language processing. We first benchmark word-based models against char-based models (those do not involve Chinese word segmentation). We run apples-to-apples comparison between these two types of models on four NLP tasks: language modeling, document classification, machine translation and sentence matching. We observe that char-based models consistently outperform word-based model. We also compare char-based models with word-char hybrid models (Yin et al., 2016; Dong et al., 2016; Yu et al., 2017), and observe that char-based models perform better or at least as good as the hybrid model, indicating that char-based models already encode sufficient semantic information.

It is also crucial to understand the inadequacy of word-based models. To this end, we perform comprehensive analyses on the behavior of word-based models and char-based models. We identify the major factor contributing to the disadvantage of word-based models, i.e., data sparsity, which in turn leads to overfitting, prevelance of OOV words, and weak domain transfer ability.

Instead of making a conclusive (and arrogant) argument that Chinese word segmentation is not necessary, we hope this paper could foster more discussions and explorations on the necessity of the long-existing task of CWS in the community, alongside with its underlying mechanisms.

2 Related Work

Since the First International Chinese Word Segmentation Bakeoff in 2003 (Sproat and Emerson, 2003) , a lot of effort has been made on Chinese word segmentation.

Most of the models in the early years are based on a dictionary, which is pre-defined and thus independent of the Chinese text to be segmented. The simplest but remarkably robust model is the maximum matching model (Jurafsky and Martin, 2014). The simplest version of it is the left-to-right maximum matching model (*maxmatch*). Starting with the beginning of a string, maxmatch chooses the longest word in the dictionary that matches the current position, and advances to the end of the matched word in the string. Different models are proposed based on different segmentation criteria (Huang and Zhao, 2007).

With the rise of statistical machine learning methods, the task of CWS is formalized as a tagging task, i.e., assigning a BEMS label to each character of a string that indicates whether the character is the start of a word(Begin), the end of a word(End), inside a word (Middel) or a single word(Single). Traditional sequence labeling models such as HMM, MEMM and CRF are widely used (Lafferty et al., 2001; Peng et al., 2004; Zhao et al., 2006; Carpenter, 2006). .

Neural CWS Models such as RNNs, LSTMs (Hochreiter and Schmidhuber, 1997) and CNNs (Krizhevsky et al., 2012; Kim, 2014) not only provide a more flexible way to incorporate context semantics into tagging models but also relieve researchers from the massive work of feature engineering. Neural models for the CWS task have become very popular these years (Chen et al., 2015b,a; Cai and Zhao, 2016; Yao and Huang, 2016; Chen et al., 2017b; Zhang et al., 2016; Chen et al., 2017c; Yang et al., 2017; Cai et al., 2017; Zhang et al., 2017). Neural representations can be used either as a set of CRF features or as input to the decision layer.

3 Experimental Results

In this section, we evaluate the effect of word segmentation in deep learning-based Chinese NLP in four tasks, language modeling, machine translation, text classification and sentence matching/paraphrase. To enforce apples-to-apples comparison, for both the word-based model and the char-based model, we use grid search to tune all important hyper-parameters such as learning rate, batch size, dropout rate, etc.

model	dimension	ppl
word	512	199.9
char	512	193.0
word	2048	182.1
char	2048	170.9
hybrid (word+char)	1024+1024	175.7
hybrid (word+char)	2048+1024	177.1
hybrid (word+char)	2048+2048	176.2
hybrid (char only)	2048	171.6

Table 3: Language modeling perplexities in different models.

3.1 Language Modeling

We evaluate the two types of models on Chinese Tree-Bank 6.0 (CTB6). We followed the standard protocol, by which the dataset was split into 80%, 10%, 10% for training, validation and test. The task is formalized as predicting the upcoming word given previous context representations. The text is segmented using Jieba.[3] An upcoming word is predicted given the previous context representation. For different settings, context representations are obtained using the char-based model and the word-based model. LSTMs are used to encode characters and words.

Results are given in Table 3. In both settings, the char-based model significantly outperforms the word-based model. In addition to Jieba, we also used the Stanford CWS package (Monroe et al., 2014) and the LTP package (Che et al., 2010), which resulted in similar findings.

It is also interesting to see results from the hybrid model (Yin et al., 2016; Dong et al., 2016; Yu et al., 2017), which associates each word with a representation and each char with a representation. A word representation is obtained by combining the vector of its constituent word and vectors of the remaining characters. Since a Chinese word can contain an arbitrary number of characters, CNNs are applied to the combination of characters vectors (Kim et al., 2016) to keep the dimensionality of the output representation invariant.

We use *hybrid (word+char)* to denote the standard hybrid model that uses both char vectors and word vectors. For comparing purposes, we also implement a pseudo-hybrid model, denoted by *hybrid (char only)*, in which we do use a word segmentor to segment the texts, but word representations

[3] https://github.com/fxsjy/jieba

are obtained only using embeddings of their constituent characters. We tune hyper-parameters such as vector dimensionality, learning rate and batch size for all models.

Results are given in Table 3. As can be seen, the char-based model not only outperforms the word-based model, but also the hybrid (word+char) model by a large margin. The hybrid (word+char) model outperforms the word-based model. This means that characters already encode all the semantic information needed and adding word embeddings would backfire. The hybrid (char only) model performs similarly to the char-based model, suggesting that word segmentation does not provide any additional information. It outperforms the word-based model, which can be explained by that the hybrid (char only) model computes word representations only based on characters, and thus do not suffer from the data sparsity issue, OOV issue and the overfitting issue of the word-based model.

In conclusion, for the language modeling task on CTB, word segmentation does not provide any additional performance boost, and including word embeddings worsen the result.

3.2 Machine Translation

In our experiments on machine translation, we use the standard Ch-En setting. The training set consists of 1.25M sentence pairs extracted from the LDC corpora.[4] The validation set is from NIST 2002 and the models are evaluated on NIST 2003, 2004, 2005, 2006 and 2008. We followed exactly the common setup in Ma et al. (2018); Chen et al. (2017a); Li et al. (2017); Zhang et al. (2018), which use top 30,000 English words and 27,500 Chinese words. For the char-based model, vocab size is set to 4,500. We report results in both the Ch-En and the En-Ch settings.

Regarding the implementation, we compare char-based models with word-based models under the standard framework of SEQ2SEQ +attention (Sutskever et al., 2014; Luong et al., 2015). The current state-of-the-art model is from Ma et al. (2018), which uses both the sentences (seq2seq) and the bag-of-words as targets in the training stage. We simply change the word-level encoding in Ma et al. (2018) to char-level encoding. For En-Ch translation, we use the same dataset to train and test both models. As in Ma et al. (2018), the dimensionality for word vectors and char vectors is set to 512.

[4]LDC2002E18, LDC2003E07, LDC2003E14, Hansards portion of LDC2004T07, LDC2004T08 and LDC2005T06.

Results for Ch-En are shown in Table 4. As can be seen, for the vanilla SEQ2SEQ +attention model, the char-based model outperforms the word-based model across all datasets, yielding an average performance boost of +0.83. The same pattern applies to the bag-of-words framework in Ma et al. (2018). When changing the word-based model to the char-based model, we are able to obtain a performance boost of +0.63. As far as we are concerned, this is the best result on this 1.25M Ch-En dataset.

Results for En-Ch are presented in Table 5. As can be seen, the char-based model outperforms the word-based model by a huge margin (+3.13), and this margin is greater than the improvement in the Ch-En translation task. This is because in Ch-En translation, the difference between word-based and char-based models is only present in the source encoding stage, whereas in En-Ch translation it is present in both the source encoding and the target decoding stage. Another major reason that contributes to the inferior performance of the word-based model is the UNK word at decoding time, We also implemented the BPE subword model (Sennrich et al., 2016b,a) on the Chinese target side. The BPE model achieves a performance of 41.44 for the Seq2Seq+attn setting and 44.35 for bag-of-words, significantly outperforming the word-based model, but still underperforming the char-based model by about 0.8-0.9 in BLEU.

We conclude that for Chinese, generating characters has the advantage over generating words in deep learning decoding.

3.3 Sentence Matching/Paraphrase

There are two Chinese datasets similar to the Stanford Natural Language Inference (SNLI) Corpus (Bowman et al., 2015): BQ and LCQMC, in which we need to assign a label to a pair of sentences depending on whether they share similar meanings. For the BQ dataset (Chen et al., 2018), it contains 120,000 Chinese sentence pairs, and each pair is associated with a label indicating whether the two sentences are of equivalent semantic meanings. The dataset is deliberately constructed so that sentences in some pairs may have significant word overlap but complete different meanings, while others are the other way around. For LCQMC (Liu et al., 2018), it aims at identifying whether two sentences have the same intention. This task is similar to but not exactly the same as the paraphrase detection task in BQ: two sentences can have different meanings but share the same intention. For exam-

TestSet	Mixed RNN	Bi-Tree-LSTM	PKI	Seq2Seq +Attn (word)	Seq2Seq +Attn (char)	Seq2Seq (word) +Attn+BOW	Seq2Seq (char) +Attn+BOW
MT-02	36.57	36.10	39.77	35.67	36.82 (+1.15)	37.70	40.14 (+0.37)
MT-03	34.90	35.64	33.64	35.30	36.27 (+0.97)	38.91	40.29 (+1.38)
MT-04	38.60	36.63	36.48	37.23	37.93 (+0.70)	40.02	40.45 (+0.43)
MT-05	35.50	34.35	33.08	33.54	34.69 (+1.15)	36.82	36.96 (+0.14)
MT-06	35.60	30.57	32.90	35.04	35.22 (+0.18)	35.93	36.79 (+0.86)
MT-08	–	–	24.63	26.89	27.27 (+0.38)	27.61	28.23 (+0.62)
Average	–	–	32.51	33.94	34.77 (+0.83)	36.51	37.14 (+0.63)

Table 4: Results of different models on the Ch-En machine translation task. Results of Mixed RNN (Li et al., 2017), Bi-Tree-LSTM (Chen et al., 2017a) and PKI (Zhang et al., 2018) are copied from the original papers.

TestSet	Seq2Seq +Attn (word)	Seq2Seq +Attn (char)	Seq2Seq +Attn+BOW	Seq2Seq (char) +Attn+BOW
MT-02	42.57	44.09 (+1.52)	43.42	46.78 (+3.36)
MT-03	40.88	44.57 (+3.69)	43.92	47.44 (+3.52)
MT-04	40.98	44.73 (+3.75)	43.35	47.29 (+3.94)
MT-05	40.87	42.50 (+1.63)	42.63	44.73 (+2.10)
MT-06	39.33	42.88 (+3.55)	43.31	46.66 (+3.35)
MT-08	33.52	35.36 (+1.84)	35.65	38.12 (+2.47)
Average	39.69	42.36 (+2.67)	42.04	45.17 (+3.13)

Table 5: Results on the En-Ch machine translation task.

ple, the meanings of "*My phone is lost*" and "*I need a new phone*" are different, but their intentions are the same: buying a new phone.

Each pair of sentences in the BQ and the LCQMC dataset is associated with a binary label indicating whether the two sentences share the same intention, and the task can be formalized as predicting this binary label. To predict correct labels, a model needs to handle the semantics of the sub-units of a sentence, which makes the task very appropriate for examining the capability of semantic models.

We compare char-based models with word-based models. For the word-based models, texts are segmented using Jieba. The SOTA results on these two datasets is achieved by the bilateral multi-perspective matching model (BiMPM) (Wang et al., 2017). We use the standard settings proposed by BiMPM, i.e. 200d word/char embeddings, which are randomly initialized.

Results are shown in Table 6. As can be seen, the char-based model significantly outperforms the word-based model by a huge margin, +1.34 on the LCQMC dataset and +2.90 on the BQ set. For this paraphrase detection task, the model needs to handle the interactions between sub-units of a sentence. We conclude that the char-based model is significantly better in this respect.

3.4 Text Classification

For text classification, we use the currently widely used benchmarks including:

- ChinaNews: Chinese news articles split into 7 news categories.
- Ifeng: First paragraphs of Chinese news articles from 2006-2016. The dataset consists of 5 news categories;
- JD_Full: product reviews in Chinese crawled from JD.com. The reviews are used to predict customers' ratings (1 to 5 stars), making the task a five-class classification problem.
- JD_binary: the same product reviews from JD.com. We label 1, 2-star reviews as "negative reviews" and 4 and 5-star reviews as "positive reviews" (3-star reviews are ignored), making the task a binary-classification problem.
- Dianping: Chinese restaurant reviews crawled from the online review website Dazhong Dianping (similar to Yelp). We collapse the 1, 2 and 3-star reviews to "negative reviews" and 4 and 5-star reviews to "positive reviews".

The datasets were first introduced in Zhang and LeCun (2017). We trained the word-based version and the char-based version of bi-directional LSTM models to solve this task. Results are shown in Table 7. As can be seen, the only dataset that the char-based model underperforms the word-based model is the chinanews dataset, but the difference is quite small (0.05). On all the other datasets, the char-based model significantly outperforms the word-based model.

Domain Adaptation Ability (Daumé III, 2007;

Dataset	description	char valid	word valid	char test	word test
LCQMC	238.7K/8.8K/12.5K	84.70	83.48	**84.43** (+1.34)	83.09
BQ	100K/10K/10K	82.59	79.63	**82.19** (+2.90)	79.29

Table 6: Results on the LCQMC and BQ corpus.

Dataset	description	char valid	word valid	char test	word test
chinanews	1260K/140K/112K	91.81	91.82	91.80	**91.85** (+0.05)
dianping	1800K/200K/500K	78.80	78.47	**78.76** (+0.36)	78.40
ifeng	720K/80K/50K	86.04	84.89	**85.95** (+1.09)	84.86
jd_binary	3600K/400K/360K	92.07	91.82	**92.05** (+0.16)	91.89
jd_full	2700K/300K/250K	54.29	53.60	**54.18** (+0.81)	53.37

Table 7: Results on the validation and the test set for text classification.

train_dianping_test_jd		
model	acc	proportion of sen containing OOV
word-based	81.28%	11.79%
char-based	83.33%	0.56%

train_jd_test_dianping		
model	acc	proportion of sen containing OOV
word-based	67.32%	7.10%
char-based	67.93%	46.85%

Table 8: Domain adaptation of the word-based model and the char-based model

Jiang, 2008; Zhuang et al., 2010) refers to the ability of extending a model learned from one data distribution (the source domain) for a different (but related) data distribution (the target domain). Because of the data sparsity issue, we hypothesize that char-based models have greater domain adaptation ability than word-based models.

We test our hypothesis on different sentiment analysis datasets. We train the word-based model and the char-based model on Dianping (2M restaurant reviews) and test the two models on jd_binary (0.25M product reviews), as denoted by train_dianping_test_jd. We also train models on jd_binary and test them on Dianping, as denoted by train_jd_test_dianping). Results are given in Table 8. As expected, the char-based model has more domain adaptation ability and performs better than the word-based model on both settings. The OOV issue is especially serious for the word-based model. In the train_dianping_test_jd setting, 11.79% of the test sentences contain OOVs for the word-based model, whereas this number is only 0.56% for the char-based model. Similar observation holds for the train_jd_test_dianping setting.

4 Analysis

In this section, we aim at understanding why word-based models underperform char-based models. We acknowledge that it is impossible to thoroughly inspect the inner mechanism of word-based models, but we try our best to identify major factors explaining the inferiority of word-based models.

4.1 Data Sparsity

A common method to avoid vocabulary size getting too big is to set a frequency threshold, and use a special UNK token to denote all words whose frequency is below the threshold. The value of the frequency threshold is closely related to the vocabulary size, and consequently the number of parameters. Figure 2 shows the correlation between the vocabulary size and the frequency threshold, along with the correlation between model performances and the frequency threshold. For both the char-based model and the word-based model, using all words/chars (threshold set to 0) leads to bad results. The explanation is intuitive: it is hard to learn the semantics of infrequent words/characters.

For the char-based model, the best performance is obtained when character frequency threshold is set to 5, resulting in a vocabulary size of 1,432 and a medium character frequency of 72. For the word-based model, the best performance is obtained when word frequency threshold is set to 50, in which case the vocabulary size is 1,355 and the medium word frequency is 83. As can be seen, the vocabulary size and the medium word frequency for the best word-based model is similar to those of the best char-based model. This means, for a given dataset, in order to learn the word/char semantics well, the model needs to have enough exposure to each word/character, the amount of which is ap-

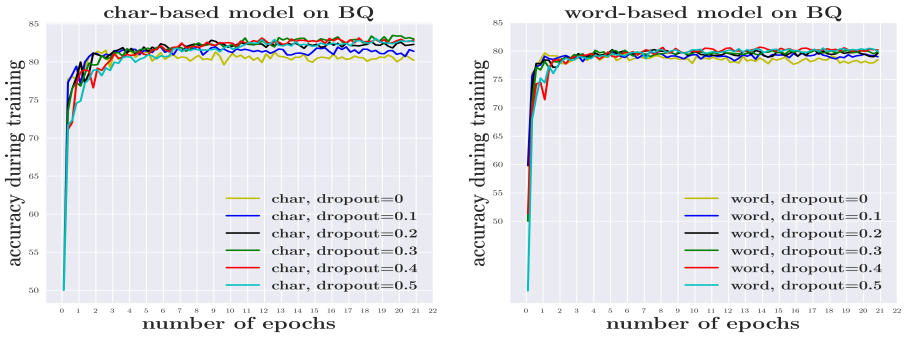

Figure 1: Effects of dropout rates on the char-based model and the word-based model.

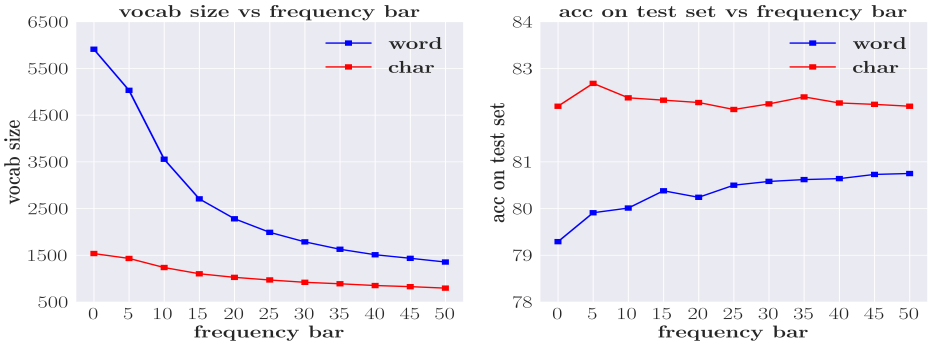

Figure 2: Effects of data sparsity on the char-based model and the word-based model.

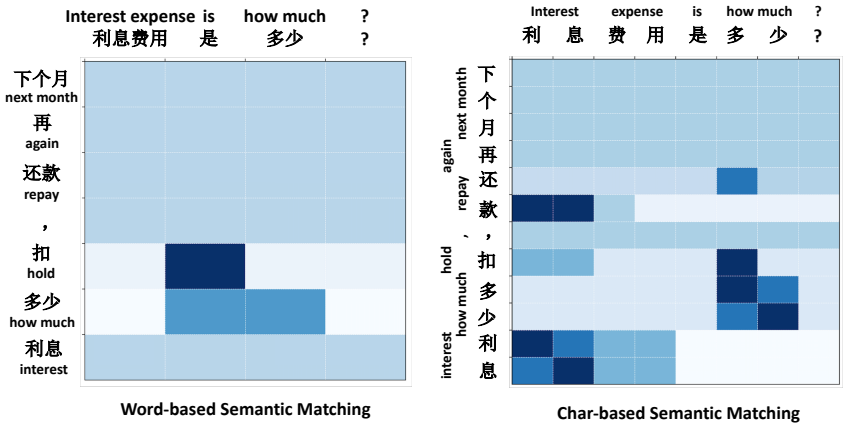

Figure 3: Semantic matching between two Chinese sentences with char-based models and word-based models.

proximately the same across different models. For the word-based model, this requirement is particularly hard to meet due to its sparseness.

4.2 Out-of-Vocabulary Words

One possible explanation for the inferiority of the word-based model is that it contains too many OOVs. If so, we should be able to narrow or even close the gap between word-based models and char-based models by decreasing the number of OOVs. As discussed in Section 4.2, setting the frequency threshold low to avoid OOVs will hinder the performance because it worsen the data sparsity issue. We thus use an alternative strategy: for different word-frequency thresholds, we remove sentences that contain word OOVs from all of the training, validation and test sets. Figure 4 shows vocabulary sizes of the training set and accuracies plotted

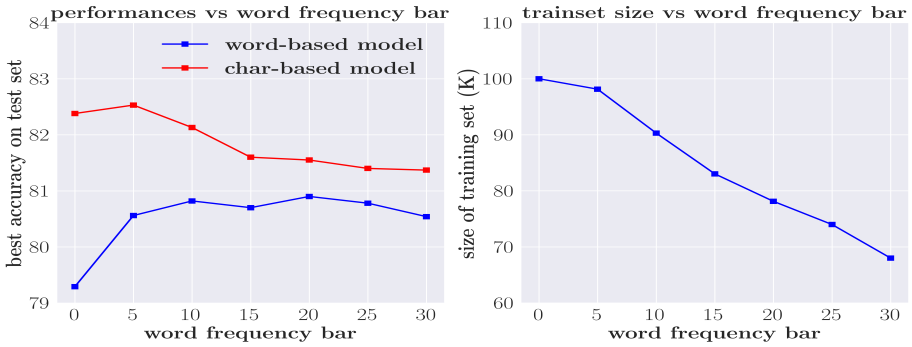

Figure 4: Effects of removing training instances containing OOV words.

against word frequency threshold. As can be seen, the gap between the two types of models is gradually narrowed as we increase the word-frequency threshold. It is also interesting that the curve for the char-based model goes up slightly at the beginning and then goes down steadily. It is because the OOV issue is not severe for the char-based model and thus does not affect the performance much. However, as we remove more and more training examples, the shrinking training dataset creates a bigger problem. By contrast, for the word-based model, the performance keeps increasing even when the frequency threshold is set to 50, meaning that the positive influence of removing some OOVs outweighs the negative influence of eliminating some training data. In conclusion, the word-based model suffers from the OOV issue. This issue can be alleviated by reducing the number of OOVs in the datasets.

4.3 Overfitting

The data sparsity issue leads to the fact that word-based models have more parameters to learn, and thus are more prone to overfitting. We conducted experiments on the BQ dataset (Chen et al., 2018) and the results validate this point (Figure 1). To achieve the best results, a larger dropout rate is needed for the word-based model (0.5) than the char-based model (0.3). This means overfitting is a severe issue for the word-based model. We also observe that curves with different dropout rates are closer together in word-based models than in char-based models, which means the dropout technique is not enough to resolve the overfitting issue. the char-based model without dropout already achieves better performance (80.82) than the word-based model with the optimal dropout rate (80.65).

4.4 Visualization

The BQ semantic matching task aims at deciding whether two sentences have the same intention. Figure 3 tangibly shows why the char-based model outperforms the word-based model. The heatmap denotes the attention matching values between tokens of two two sentences, computed by the BiPMP model (Wang et al., 2017). The input two sentences are: (1) 利息费用是多少 (how much is the interest expense), with segmented text being 利息费用 (interest expense) 是 (is) 多少 (how much) and (2) 下一个月还款要扣多少利息 (how much interest do I have to pay if I repay the bill next month), with segmented text being 下个月(next month) 还款 (repay), 扣 (hold) 多少 (how much) 利息 (interest). For word-based semantic matching, since 利息费用(interest expense) is treated as a single word, it fails to be mapped to 利息(interest). This is not the case with the char-based model since the same character in the two sentences are more easily mapped.

5 Conclusion

In this paper, we ask the fundamental question of whether word segmentation is necessary for deep learning of Chinese representations. We benchmark such word-based models against char-based models in four end-to-end NLP tasks, and enforce apples-to-apples comparisons as much as possible. We observe that char-based models consistently outperform word-based models. Building upon these findings, we show that word-based models' inferiority is due to the sparseness of word distributions, which leads to more out-of-vocabulary words, overfitting and lack of domain generalization ability. We hope this paper will foster more discussions on the necessity of the long-existing task of CWS in the community.

References

Xuejun Bai, Guoli Yan, Simon P Liversedge, Chuanli Zang, and Keith Rayner. 2008. Reading spaced and unspaced chinese text: Evidence from eye movements. *Journal of Experimental Psychology: Human Perception and Performance*, 34(5):1277.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *EMNLP*.

Deng Cai and Hai Zhao. 2016. Neural word segmentation learning for chinese. *ACL*.

Deng Cai, Hai Zhao, Zhisong Zhang, Yuan Xin, Yongjian Wu, and Feiyue Huang. 2017. Fast and accurate neural word segmentation for chinese. *ACL*.

Bob Carpenter. 2006. Character language models for chinese word segmentation and named entity recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 169–172.

Wanxiang Che, Zhenghua Li, and Ting Liu. 2010. Ltp: A chinese language technology platform. In *Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations*, pages 13–16. Association for Computational Linguistics.

Huadong Chen, Shujian Huang, David Chiang, and Jiajun Chen. 2017a. Improved neural machine translation with a syntax-aware encoder and decoder. *ACL*.

Jing Chen, Qingcai Chen, Xin Liu, Haijun Yang, Daohe Lu, and Buzhou Tang. 2018. The bq corpus: A large-scale domain-specific chinese corpus for sentence semantic equivalence identification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4946–4951.

Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2017b. A feature-enriched neural model for joint chinese word segmentation and part-of-speech tagging. *IJCAI*.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, and Xuanjing Huang. 2015a. Gated recursive neural network for chinese word segmentation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1744–1753.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015b. Long short-term memory neural networks for chinese word segmentation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1197–1206.

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

Hal Daumé III. 2007. Frustratingly easy domain adaptation. *ACL*.

Chuanhai Dong, Jiajun Zhang, Chengqing Zong, Masanori Hattori, and Hui Di. 2016. Character-based lstm-crf with radical-level features for chinese named entity recognition. In *Natural Language Understanding and Intelligent Applications*, pages 239–250. Springer.

Schubert Foo and Hui Li. 2004. Chinese word segmentation and its effect on information retrieval. *Information processing & management*, 40(1):161–190.

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

Changning Huang and Hai Zhao. 2007. Chinese word segmentation: A decade review. *Journal of Chinese Information Processing*, 21(3):8–20.

Jing Jiang. 2008. Domain adaptation in natural language processing. Technical report.

Dan Jurafsky and James H Martin. 2014. *Speech and language processing*, volume 3. Pearson London.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *EMNLP*.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *AAAI*, pages 2741–2749.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Junhui Li, Deyi Xiong, Zhaopeng Tu, Muhua Zhu, Min Zhang, and Guodong Zhou. 2017. Modeling source syntax for neural machine translation. *arXiv preprint arXiv:1705.01020*.

Wei Liu, Ben Allison, David Guthrie, and Louise Guthrie. 2007. Chinese text classification without automatic word segmentation. In *Sixth International Conference on Advanced Language Processing and Web Information Technology (ALPIT 2007)*, pages 45–50. IEEE.

Xin Liu, Qingcai Chen, Chong Deng, Huajun Zeng, Jing Chen, Dongfang Li, and Buzhou Tang. 2018. Lcqmc: A large-scale chinese question matching corpus. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1952–1962.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *ACL*.

Shuming Ma, Xu Sun, Yizhong Wang, and Junyang Lin. 2018. Bag-of-words as target for neural machine translation. *arXiv preprint arXiv:1805.04871*.

Will Monroe, Spence Green, and Christopher D Manning. 2014. Word segmentation of informal arabic with domain adaptation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 206–211.

Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *Proceedings of the 20th international conference on Computational Linguistics*, page 562. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Edinburgh neural machine translation systems for wmt 16. *WMT*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. *ACL*.

Richard Sproat and Thomas Emerson. 2003. The first international chinese word segmentation bakeoff. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*, pages 133–143. Association for Computational Linguistics.

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

Jie-Li Tsai and George W McConkie. 2003. Where do chinese readers send their eyes? In *The Mind's Eye*, pages 159–176. Elsevier.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. *IJCAI*.

Fei Xia. 2000. The part-of-speech tagging guidelines for the penn chinese treebank (3.0). *IRCS Technical Reports Series*, page 38.

Jia Xu, Richard Zens, and Hermann Ney. 2004. Do we need chinese word segmentation for statistical machine translation? In *Proceedings of the Third SIGHAN Workshop on Chinese Language Processing*.

Jie Yang, Yue Zhang, and Fei Dong. 2017. Neural word segmentation with rich pretraining. *ACL*.

Jie Yang, Yue Zhang, and Shuailong Liang. 2018. Subword encoding in lattice lstm for chinese word segmentation. *arXiv preprint arXiv:1810.12594*.

Yushi Yao and Zheng Huang. 2016. Bi-directional lstm recurrent neural network for chinese word segmentation. In *International Conference on Neural Information Processing*, pages 345–353. Springer.

Rongchao Yin, Quan Wang, Peng Li, Rui Li, and Bin Wang. 2016. Multi-granularity chinese word embedding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 981–986.

Jinxing Yu, Xun Jian, Hao Xin, and Yangqiu Song. 2017. Joint embeddings of chinese words, characters, and fine-grained subcharacter components. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 286–291.

Shiwen Yu, Jianming Lu, Xuefeng Zhu, Huiming Duan, Shiyong Kang, Honglin Sun, Hui Wang, Qiang Zhao, and Weidong Zhan. 2001. Processing norms of modern chinese corpus. Technical report, Technical report.

Hua-Ping Zhang, Hong-Kui Yu, De-Yi Xiong, and Qun Liu. 2003. Hhmm-based chinese lexical analyzer ictclas. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*, pages 184–187. Association for Computational Linguistics.

Jiacheng Zhang, Yang Liu, Huanbo Luan, Jingfang Xu, and Maosong Sun. 2018. Prior knowledge integration for neural machine translation using posterior regularization. *arXiv preprint arXiv:1811.01100*.

Meishan Zhang, Guohong Fu, and Nan Yu. 2017. Segmenting chinese microtext: Joint informal-word detection and segmentation with neural networks. In *IJCAI*, pages 4228–4234.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Transition-based neural word segmentation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 421–431.

Xiang Zhang and Yann LeCun. 2017. Which encoding is the best for text classification in chinese, english, japanese and korean? *arXiv preprint arXiv:1708.02657*.

Hai Zhao, Chang-Ning Huang, and Mu Li. 2006. An improved chinese word segmentation system with conditional random field. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 162–165.

Hai Zhao, Masao Utiyama, Eiichiro Sumita, and Bao-Liang Lu. 2013. An empirical study on word segmentation for chinese machine translation. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 248–263. Springer.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for chinese word segmentation and pos tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 647–657.

Hao Zhou, Zhenting Yu, Yue Zhang, Shujian Huang, XIN-YU DAI, and Jiajun Chen. 2017. Word-context character embeddings for chinese word segmentation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 760–766.

Fuzhen Zhuang, Ping Luo, Hui Xiong, Yuhong Xiong, Qing He, and Zhongzhi Shi. 2010. Cross-domain learning from multiple sources: A consensus regularization perspective. *IEEE Transactions on Knowledge and Data Engineering*, 22(12):1664–1678.