MarkBERT: Marking Word Boundaries Improves Chinese BERT

Anonymous ACL submission

Abstract

We present a Chinese BERT model dubbed MarkBERT that uses word information in this work. Existing word-based BERT models regard words as basic units, however, due to the vocabulary limit of BERT, they only cover high-frequency words and fall back to character level when encountering out-of-vocabulary (OOV) words. Different from existing works, MarkBERT keeps the vocabulary being Chinese characters and inserts boundary markers between contiguous words. Such design enables the model to handle any words in the same way, no matter they are OOV words or not. Besides, our model has two additional benefits: first, it is convenient to add word-level learning objectives over markers, which is complementary to traditional character and sentence-level pretraining tasks; second, it can easily incorporate richer semantics such as POS tags of words by replacing generic markers with POS tag-specific markers. MarkBERT pushes the state-of-the-art of Chinese named entity recognition from 95.4% to 96.5% on the MSRA dataset and from 82.8% to 84.2% on the OntoNotes dataset, respectively. Compared to previous word-based BERT models, MarkBERT achieves better accuracy on text classification, keyword recognition, and semantic similarity tasks.[1]

1 Introduction

Chinese words can be composed of multiple Chinese characters. For instance, the word 地球 (earth) is made up of two characters 地 (ground) and 球 (ball). However, there are no delimiters (i.e., space) between words in written Chinese sentences. Traditionally, word segmentation is an important first step for Chinese natural language processing tasks (Chang et al., 2008). Instead, with the rise of pretrained models (Devlin et al., 2018), Chinese BERT models are dominated by character-based ones (Cui et al., 2019a; Sun et al., 2019; Cui et al., 2020; Sun et al., 2021b,a), where a sentence is represented as a sequence of characters. There are several attempts at building Chinese BERT models where word information is considered. Existing studies tokenize a word as a basic unit (Su, 2020), as multiple characters (Cui et al., 2019a) or a combination of both (Zhang and Li, 2020; Lai et al., 2021; Guo et al., 2021). However, due to the limit of the vocabulary size of BERT, these models only learn for a limited number (e.g., 40K) of words with high frequency. Rare words below the frequency threshold will be tokenized as separate characters so that the word information is neglected.

In this work, we present a simple framework, MarkBERT, that considers Chinese word information. Instead of regarding words as basic units, we use character-level tokenizations and inject word information via inserting special markers between contiguous words. The occurrence of a marker gives the model a hint that its previous character is the end of a word and the following character is the beginning of another word. Such a simple model design has the following advantages. First, it avoids the problem of OOV words since it deals with common words and rare words (even the words never seen in the pretraining data) in the same way. Second, the introduction of marker allows us to design word-level pretraining tasks (such as replaced word detection illustrated in section 3), which are complementary to traditional character-level pretraining tasks like masked language modeling and sentence-level pretraining tasks like next sentence prediction. Third, the model is easy to be extended to inject richer semantics of words. For example, we can inject information such as POS tags into pretrained model by simply replacing the generic word marker `[S]` with POS tag-specific markers (e.g., $[\mathrm{S}_{NN}]$ for markers of nouns and $[\mathrm{S}_{VV}]$ for markers of verbs) as illustrated in Figure 1.

[1] All the codes and models will be made publicly available at https://github.com/

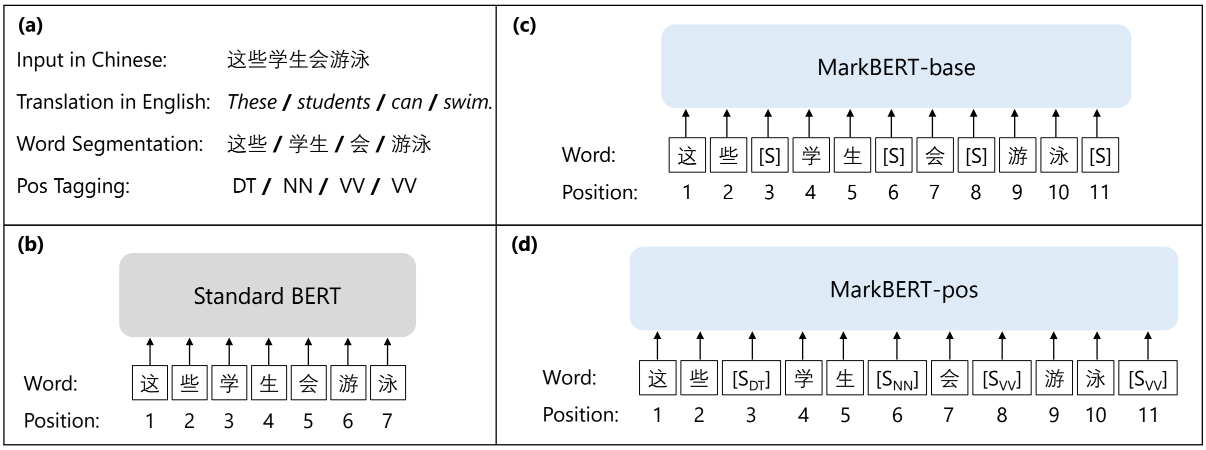

Figure 1: An illustrative example of our model. Box (a) gives the original input written in Chinese, its translation in English, word segmentation results given by an off-the-shell text analyzer, and the POS tags of words. Box (b) shows a traditional character-level Chinese BERT. Box (c) shows the base model of MarkBERT, in which generic word boundary markers [S] are inserted between words. In box (d), the POS tag version of MarkBERT replaces the generic markers [S] with POS tag specific ones such as $[S_{NN}]$ and $[S_{VV}]$.

In the pretraining stage, we force the markers to understand the contexts around them while serving as separators between words. We train our model with two pretraining tasks. The first task is masked language modeling. We also mask markers such that word boundary knowledge can be learned. The second task is replaced word detection. We replace a word with an artificially generated one, take replaced contextual representation of the marker following the word, and ask the model to distinguish whether the marker follows a correct word or not.

On the task of named entity recognition (NER), we demonstrate that MarkBERT achieves the new state-of-the-art on both MSRA and OntoNotes datasets (Huang et al., 2015; Zhang and Yang, 2018), surpassing previous systems. Compared with other word-level Chinese BERT models, we show that MarkBERT performs better on text classification, keyword recognition, and semantic similarity tasks. We summarize the major contributions of this work as follows.

- We present a simple and effective Chinese pretrained model MarkBERT that considers word information without aggravating the problem of OOV words.
- We demonstrate that our model achieves state-of-the-art performance on Chinese NER while performs better than previous word-based Chinese BERT models on three natural language understanding tasks.

2 Related Work

We describe related work on injecting word information to Chinese BERT and the use of marker in natural language understanding tasks.

2.1 Chinese BERT

Pre-trained models exemplified by BERT (Devlin et al., 2018) and RoBERTa (Cui et al., 2019a) have been proved successful in various Chinese NLP tasks (Xu et al., 2020; Cui et al., 2019b). Existing Chinese BERT models that incorporate word information can be divided into two categories. The first category uses word information in the pretraining stage but represents a text as a sequence of characters when the pretrained model is applied to downstream tasks. For example, Cui et al. (2019a) use the whole-word-masking strategy that masks word spans and predicts continuously multiple masked positions. Lai et al. (2021) incorporate lexicon information by concatenating the lexicons along with character-level context. The second category uses word information when the pretrained model is used in downstream tasks. For example, Su (2020) uses a word-level vocabulary instead of characters. If a word 地球 is included in the vocabulary, its constitutes 地 and 球 will not be considered as input tokens. Zhang and Li (2020) go one step further by constructing two independent encoders that encode character-level and word-level information separately and concatenate them at the top layers of two encoders. Similarly, Guo et al. (2021) encode both character-level and word-level information.

They move the information aggregation stage to the embedding level.

2.2 Marker Insertion in NLU

The idea of inserting markers is explored in entity-related natural language understanding tasks, especially in relation classification. Given a subject entity and an object entity as the input, existing work inject untyped markers (Sun et al., 2019; Soares et al., 2019) or entity-specific markers (Zhong and Chen, 2020) around the entities, and make better predictions of the relations of the entities.

3 MarkBERT Pre-training

In this section, we first introduce the background of character level Chinese pre-trained models; then we introduce the structure of our MarkBERT model. After describing the structure of MarkBERT, we introduce the training process of the MarkBERT. In addition, MarkBERT can be extended with rich semantics such as pos-tags, therefore we introduce a MarkBERT-POS model. Finally, we provide details of the entire training process.

3.1 Character Level Chinese BERT

In language model pre-training, BERT (Devlin et al., 2018) first introduced the masked language modeling strategy to learn the context information by replacing tokens with masks and assign the model to predict the masked tokens based on the contexts around them using the self-attention transformers structure (Vaswani et al., 2017). In Chinese language model pre-training, the encoding unit is different from the widely used BPE encoding in English: Chinese pre-trained models are usually character-level and word level information is typically neglected.

3.2 MarkBERT Model

To make better use of word-level information in Chinese pre-training, we introduce a simple framework called MarkBERT. We insert markers between word spans to give explicit boundary information for the model pre-training.

As seen in Figure 1, we first use a segmentation tool to obtain word segmentations, then we insert special markers between word spans as separators between characters. These markers are treated as normal characters so they take positions in the transformers structure. Plus, they can also be masked for the mask language modeling task to predict, therefore the encoding process needs to be aware of predicting word boundaries rather than simply filling in masks from the context. The mask prediction task becomes more challenging since predicting the masks correctly requires a better understanding of the word boundaries. In this way, the model is still character-level encoded while it is aware of word boundaries since word-level information is given explicitly.

3.3 Replaced Word Detection

Inserting special markers allows the pre-trained model to recognize word boundaries while maintaining a character-level model. Further, these special markers can be used to construct a word-level pre-training task which can be complementary to the character-level masked language modeling task.

We construct a replaced word detection task as an auxiliary task to the masked language modeling task. We construct a bipolar classification task that detects whether the word span is replaced by a confusion word. Specifically, given a word span, we take the representations of the marker after it and make binary prediction.

When a word span is replaced by a confusion word, as seen in Figure 2, the marker is supposed to make a "replaced" prediction labeled as "False". When the word spans are not changed, the marker will make an "unchanged" prediction labeled as "True". Therefore, suppose the representation of the i^{th} marker is x^i with label y^{true} and y^{false}, the replaced word detection loss is:

$$\mathcal{L} = -\sum_i [y^{true} \cdot log(x_y^i) + y^{false} \cdot log(x_y^i)] \quad (1)$$

We add this loss term to the masked language modeling loss as a multi task training process.

The construction of the confusions could be various. We adopt two simple strategies: (1) we use synonyms as confusions; (2) we use words that are similar in phonetics (pinyin) in Chinese. To obtain the synonyms, we use an external word embedding provided by Zhang and Yang (2018). We calculate the cosine similarity between words and use the most similar ones as the synonyms confusions. To obtain the phonetic-based confusions, as seen in Figure 2, we use an external tool to get the phonetics of the word and select a word that share the same phonetics as its confusions.

In this way, the markers can be more sensitive to the word span in the context since these markers

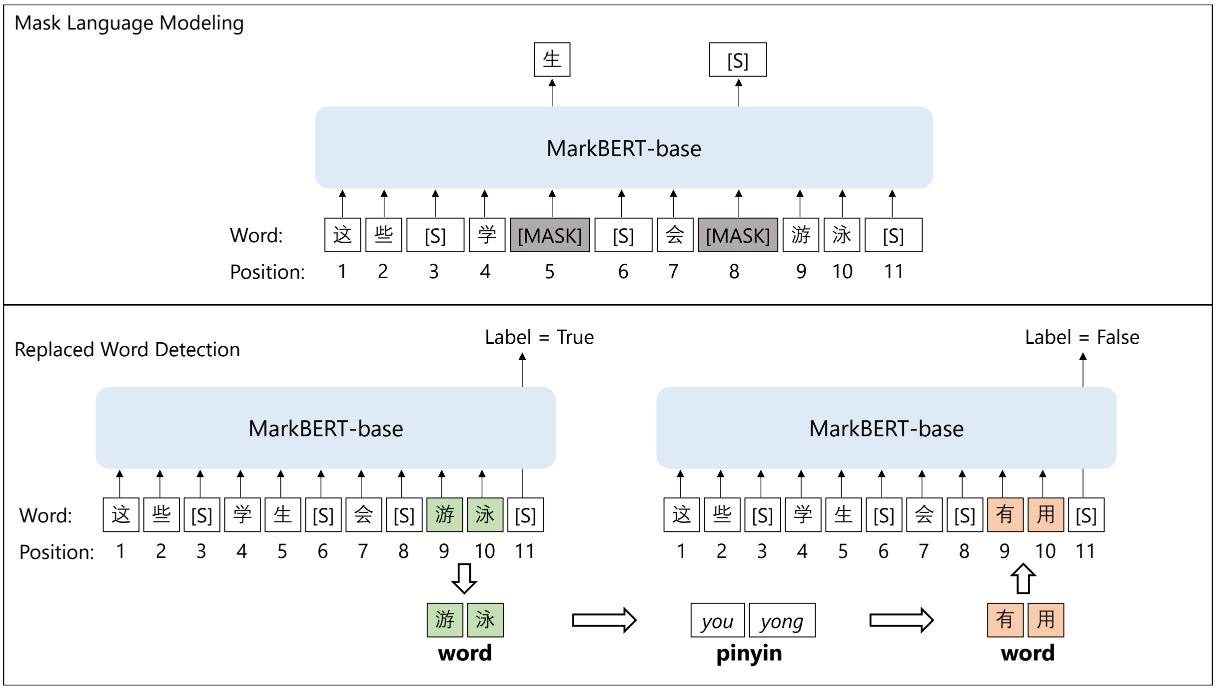

Figure 2: Illustration of the predicting tasks of Masked Language Modeling and Replaced Word Detection. Here, `[S]` is the inserted markers.

are assigned to discriminate the representation type of the word spans before them. This process is similar to an ELECTRA (Clark et al., 2020) framework. MarkBERT uses the inserted markers to run the discrimination process inside the encoder and use external confusions instead of using another generator to build texts for the discriminator.

3.4 MarkBERT-POS Model

In addition, we can improve the MarkBERT model by using richer-semantic markers instead of simply a special token. Similar to the MarkBERT model, we use part-of-speech tags as markers to insert between word spans to construct a richer-semantic model called MarkBERT-POS. In this way, the model can be given clearer information for understanding the context. As seen in Figure 1(d), these special markers can be replaced by POS-tags acquired externally. These POS-tag markers can also be masked as well, so the mask language modeling task also needs to predict correct POS-tags.

The idea of using pos-tags is a naive usage of expanding markers. Our model can be further expanded with more helpful information as the inserted special markers.

3.5 Pre-Training

The pre-training process is a multi task framework consisting of mask language modeling task and replaced word detection task.

In the masked language modeling task, we employ both the masked language modeling strategy and the whole-word-masking strategy. In the replaced word detection task, as seen in Figure 2, when the word span is replaced by confusion words, the model is supposed to correct the confusions. This correction process is similar to MacBERT (Cui et al., 2020). For the confusion generation, we use synonyms and pinyin-based confusions. The synonyms are obtained by a synonym dictionary based on calculating the cosine similarity between the Chinese word-embeddings provided by Zhang and Yang (2018).

In our MarkBERT pre-training, the mask ratio is still 15% of the total characters. For 30% of the time, we do not insert any markers so that the model can also be used in a no-marker setting which is the vanilla BERT-style model. For 50% of the time we run a whole-word-mask prediction and for the rest we run a traditional masked language model prediction. In the marker insertion, for 30% of the time, we replace the word span with a phonetic(pinyin)-based confusion or a synonym-based confusion word and the marker

	MSRA(Test)			OntoNotes(Dev)			OntoNotes(Test)		
	Acc.	Recall	F1	Acc.	Recall	F1	Acc.	Recall	F1
BERT (Devlin et al., 2018)	94.9	94.1	94.5	74.8	81.8	78.2	78.0	75.7	80.3
RoBERTa (Cui et al., 2019a)	95.3	94.9	95.1	76.8	80.7	78.7	77.6	83.5	80.5
FLAT-BERT (Li et al., 2020)	-	-	96.1	-	-	-	-	-	81.8
Soft-Lexicon (Ma et al., 2019)	95.8	95.1	95.4	-	-	-	83.4	82.2	82.8
RoBERTa (ours)	95.7	94.8	95.2	80.3	76.4	78.3	78.8	83.4	81.1
MarkBERT (ours)	**96.5**	**96.5**	**96.5**	**84.1**	**83.5**	**83.8**	**83.5**	**85.4**	**84.2**

Table 1: NER results on the MSRA and OntoNotes dataset.

will predict a phonetic(pinyin)-confusion marker or a synonym-confusion marker; for the rest of the time, the marker will predict a normal-word marker.

We need to notice that most of the time the marker is a normal marker if the normal markers are not POS-tag enhanced. Therefore, we only calculate 15 % percent of loss on these normal markers to avoid imbalance labels of the marker learning process. During fine-tuning on downstream tasks, we use the markers in the input texts. Also, we can save the markers and downgrade the model to a vanilla BERT-style model for easier usage. We give implementation details in the appendix.

4 Experiments

To test the performance of our proposed MarkBERT, we conduct experiments on the NER and other natural language understanding tasks.

4.1 NER Task

In the NER task, we use the MSRA (Levow, 2006) and Ontonotes (Weischedel et al., 2013) datasets with the same data-split as in Ma et al. (2019) and Li et al. (2020).

We establish several strong baselines to explore the effectiveness of our MarkBERT. In language understanding tasks, we compare with the RoBERTa-wwm-ext (Cui et al., 2019a) baseline, which is a whole-word-mask trained Chinese pre-trained models. We also further pre-train the RoBERTa model denoted as RoBERTa (ours) and the WoBERT model denoted as WoBERT (ours) based on our collected data which is the same data used in pre-training MarkBERT to make fair comparisons with our model. In the NER task, we compare with FLAT-BERT (Li et al., 2020) and Soft-Lexicon (Ma et al., 2019) which are state-of-the-art models on the NER task which incorporate lexicons in the transformers/LSTM structure.

4.2 Language Understanding Task

We also conduct experiments on language understanding tasks. We use various types of tasks from the CLUE benchmark (Xu et al., 2020). We use classification tasks such as TNEWS, IFLYTEK; semantic similarity task (AFQMC); coreference resolution task(WSC); keyword recognition (CSL); natural language inference task (OCNLI).

Besides the BERT-style baselines used in the NER task, we also use the word-level information enhanced models as baselines to make comparisons in the language understanding tasks. We use:

- WoBERT (Su, 2020): a word-level Chinese pre-trained model initialized from the BERT BASE pre-trained weights. It has a 60k expanded vocabulary containing commonly used Chinese words.

- AMBERT (Zhang and Li, 2020): a multi-granularity Chinese pre-trained model with two separated encoders for words and characters. The encoding representation is the character-level representation concatenated by the word-level representation;

- LICHEE (Guo et al., 2021): a multi-granularity Chinese pre-trained model that incorporates word and character representations at the embedding level.

- Lattice-BERT (Lai et al., 2021): the state-of-the-art multi-granularity model that uses lexicons as word-level knowledge concatenated to the original input context.

4.3 Downstream Task Implementations

We use the Huggingface Transformers (Wolf et al., 2020) to implement all experiments.

For the NER task, we follow the implementation details given in the Transformers toolkit. [2] For the language understanding tasks, we follow the implementation details used in the CLUE benchmark official website and the fine-tuning hyper-parameters

[2]https://github.com/huggingface/transformers

	Datasets					
	TNEWS	IFLYTEK	AFQMC	OCNLI	WSC	CSL
DEVELOPMENT						
BERT (Devlin et al., 2018)	56.09	60.37	74.10	74.70	79.22	81.02
RoBERTa (Cui et al., 2019a)	57.51	60.80	73.80	75.01	82.20	81.22
RoBERTa (ours)	57.95	60.85	74.58	75.32	84.02	81.85
WoBERT (ours)	57.01	61.10	72.80	75.00	82.72	-
MarkBERT (ours)	**58.40**	60.68	**74.89**	**75.88**	**84.60**	-
TEST						
BERT (Devlin et al., 2018)	56.58	60.29	73.70	-	62.00	80.36
RoBERTa (Cui et al., 2019a)	56.94	60.31	74.04	-	67.80	81.00
AMBERT (Zhang and Li, 2020)	-	59.73	73.86	-	78.27	85.70
LICHEE (Guo et al., 2021)	-	60.94	73.65	-	81.03	84.51
BERT (Lai et al., 2021)	-	62.20	74.00	-	79.30	81.60
Lattice-BERT (Lai et al., 2021)	-	**62.90**	74.80	-	**82.40**	84.00
RoBERTa (ours)	57.42	61.00	73.63	72.67	79.86	81.83
MarkBERT (ours)	**58.05**	62.57	**74.87**	**73.06**	81.72	**85.73**

Table 2: Evaluation results on the language understanding tasks.

used in Lattice-BERT (Lai et al., 2021).

In the NER task, we use the marker-inserted inputs in the MarkBERT since we intend to incorporate the word boundary information in recognizing entities. We use the model with the best development performance to obtain the test set result. We make a thorough discussion on this topic in the later section. For the TNEWS task, we run the raw classification results without using the keywords augmentation which is no longer a natural context. For the IFLYTEK task, we split the context and use the average of the split texts prediction since the average sequence exceeds the max sequence length. We leave the experiment results '-' if they are not listed in the official website. [3]

4.4 Results on NER Task

In Table 1, our proposed boundary-aware MarkBERT outperforms all baseline models including pre-trained models and lexicon-enhanced models.

Compared with the baseline methods, our proposed MarkBERT with markers inserted between words can lift performances by a large margin. When we insert markers using the same tokenization process used in pre-training MarkBERT in fine-tuning the MarkBERT in the NER task, we obtain a considerable performance improvement, indicating that the inserted markers catch some important fine-grained information that helps improve entity understanding. Further, when compared with previous state-of-the-art methods such as Soft-Lexicon (Ma et al., 2019) and FLAT (Li et al., 2020) which use a combination of lexicon-enhanced LSTMs/transformers and BERT, our model can also achieve higher performance. The improvement proves the effectiveness of inserting markers for better understanding word boundaries while maintaining the character-level encoding unit. In addition, we use the pos-tags as markers in the NER task and find out that the performance is slightly better than normal markers (0.1 points improvements on the F1 score on both MSRA and OntoNotes datasets), indicating that pos-tag information can be helpful but not by a large margin.

4.5 Results on Language Understanding

Table 2 shows that comparing with the RoBERTa model that uses the same pre-training data, MarkBERT is superior in all tasks. This indicates that the learned representations contain more useful information for the downstream task fine-tuning. The word-level model WoBERT (ours) trained with the same data used in MarkBERT only achieves a slightly higher accuracy in the IFLYTEK dataset which might because the IFLYTEK dataset contains very long texts where word-level model is superior since it can process more contexts while the total sequence lengths of character level and word level model are both 512.

When comparing with previous works that focus on word-level information, MarkBERT achieves higher performances than the multi-grained encoding method AMBERT as well as LICHEE which

[3] https://github.com/CLUEbenchmark/CLUE

	Datasets				
	MSRA	Ontonotes	TNEWS	IFLYTEK	AFQMC
DEVELOPMENT	F1	F1	Acc.	Acc.	Acc.
MarkBERT	**96.5**	**83.8**	**58.4**	60.6	**74.8**
MarkBERT-rwd-pho	96.2	83.4	58.0	60.8	74.3
MarkBERT-rwd-syn	96.2	83.5	58.0	60.9	74.5
MarkBERT-MLM	96.0	83.3	58.0	60.7	74.6
MarkBERT-w/o marker	95.5	79.2	58.2	**61.0**	74.5
RoBERTa (ours)	95.1	78.2	57.9	60.8	74.5

Table 3: Ablation Studies on the NER and the language understanding tasks using dev set results.

incorporates word information as an additional embedding. We can assume that adding word-level information through *horizontal* markers is more effective than *vertically* concatenating word-level information. When comparing with the LatticeBERT model, our method can still reach a competitive level of performance, meanwhile the relative improvements of our model is larger than the improvements of the LatticeBERT model. Please note that the lexicons used in LatticeBERT training actually contains more segmentation possibilities which can significantly increase the downstream task performance over the word segmentation based methods (Zhang and Yang, 2018). The basic idea of incorporating lexicons is parallel with the marker insertion framework. MarkBERT makes use of word-level information in a different perspective.

4.6 Model Analysis

In this section, we conduct ablation experiments to explore the effectiveness of each parts in our MarkBERT framework in different tasks. We test different variants of MarkBERT:

- MarkBERT-MLM only considers the MLM task without the replaced word detection task; the masked language model will predict masked tokens as well as inserted markers.

- MarkBERT-rwd is a version that removes phonetics words or synonyms separately in the replaced word detection process.

- MarkBERT-w/o marker is a version that removed markers which is the same as the vanilla BERT model.

4.6.1 MarkBERT-MLM without RWD

To explore which parts in MarkBERT is more effective, we conduct an experiment as seen in Table 3. We only use the masked language modeling task while inserting markers without using the replaced word detection task. The model only considers inserted markers and masked language modeling tasks, while the markers will be masked and predicted as well.

As seen, the MarkBERT -MLM model gains significant boost in the NER task, indicating that word boundary information is important in the fine-grained task.

In the CLUE benchmark, the situation becomes different: in the IFLYTEK task, inserting markers will hurt the model performance which is because the sequence length exceeds the maximum length of the pre-trained model. Therefore, inserting markers will results in a lost of contexts. Generally, inserting markers is important in downstream task fine-tuning. The explicit word boundary information helps MarkBERT learn better contextualized representations.

4.6.2 Replaced Word Detection

We also test the effectiveness of the additional replaced word detection task. Specifically, we separate two confusion strategies and use phonetics and synonyms confusions solely.

As seen in Table 3, When the marker learning only includes phonetic (pinyin) confusions, the performances in the fine-tuning tasks are similar with the MarkBERT -MLM model, indicating that the phonetic confusions have a slight improvement based on the inserted markers. When the word spans are replaced by synonyms only, the performances are slightly lower than using both phonetic and synonym confusions, indicating that augmentation using various types of confusions is helpful.

4.6.3 MarkBERT -w/o marker

Further, without inserting markers, MarkBERT-w/o marker can still achieve similar performances with the baseline methods in the language modeling tasks, indicating that MarkBERT can also be used as a vanilla BERT model for easy usage in language understanding tasks. As for the NER task, inserting markers is still important, indicating that

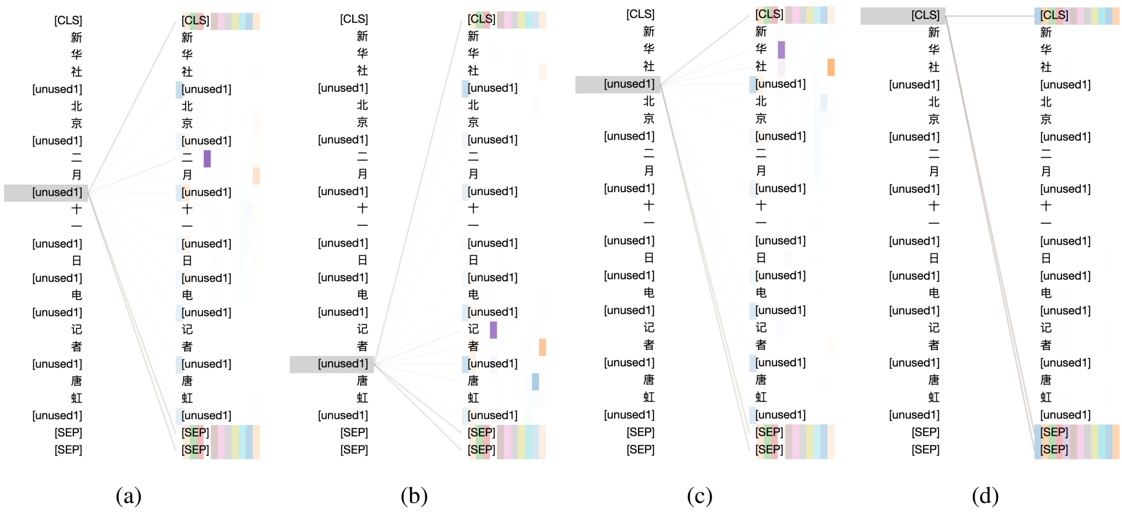

Figure 3: Visualization of attentions of the markers selected from a random layer. We use `[unused1]` in the BERT vocabulary as the inserted marker.

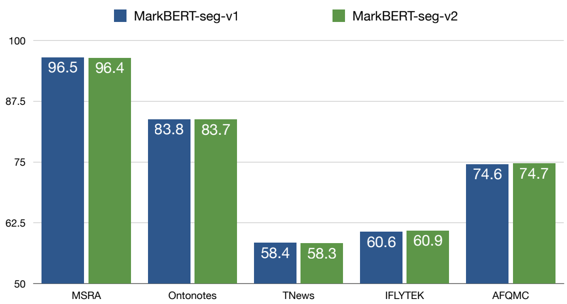

Figure 4: Results on different MarkBERT versions.

MarkBERT structure is effective in learning word boundaries for tasks that requires such fine-grained representations.

4.6.4 Visualization of Marker Attentions

To further explore how the markers work in the encoding process, we use the attention visualization tool to show the attention weights of the inserted markers. We explore the attention weights on the pre-trained MarkBERT and the fine-tuned model based on the Ontonotes NER task. As seen in Figure 3, the pre-trained representations of the markers are focusing on the local semantics of the word-level information. These markers are also connected to other special tokens indicating that the markers play important roles in learning the context representations. Further, the special tokens are the mostly focused as seen in 3 (d).

4.6.5 Influence of Different Sementation Tools in MarkBERT

The quality of the pre-processed segmentation results may play a vital role, therefore, we use a different version of segmentation in the Texsmart toolkit (Zhang et al., 2020) where the segmentations are more fine-grained to train a MarkBERT-seg-v2 model as a comparison.

As seen in figure 4, segmentation quality is trivial to MarkBERT. The performances of MarkBERT (seg-v1) is similar to a variant MarkBERT-seg-v2 using a different segmentation tool, which indicates that the training framework helps rather than the information from an external segmentation tool.

Combined with results in Table 3, we can conclude that introducing segmentation tools and use mark-style encoding is important while the quality of the segmentation is trivial.

5 Conclusion and Future Work

In this paper, we have introduced MarkBERT, a simple framework for Chinese language model pre-training. We insert special markers between word spans in the character-level encodings in pre-training and fine-tuning to make use of word-level information in Chinese. We test our proposed model on the NER tasks as well as natural language understanding tasks. Experiments show that MarkBERT makes significant improvements over baseline models. In the future, we are hoping to incorporate more information to the markers based on the simple structure of MarkBERT.

References

Pi-Chuan Chang, Michel Galley, and Christopher D Manning. 2008. Optimizing chinese word segmentation for machine translation performance. In *Proceedings of the third workshop on statistical machine translation*, pages 224–232.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for Chinese natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 657–668, Online. Association for Computational Linguistics.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019a. Pre-training with whole word masking for chinese bert. *arXiv preprint arXiv:1906.08101*.

Yiming Cui, Ting Liu, Wanxiang Che, Li Xiao, Zhipeng Chen, Wentao Ma, Shijin Wang, and Guoping Hu. 2019b. A span-extraction dataset for Chinese machine reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5886–5891, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Weidong Guo, Mingjun Zhao, Lusheng Zhang, Di Niu, Jinwen Luo, Zhenhua Liu, Zhenyang Li, and Jianbo Tang. 2021. Lichee: Improving language model pre-training with multi-grained tokenization. In *FINDINGS*.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Yuxuan Lai, Yijia Liu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2021. Lattice-bert: Leveraging multi-granularity representations in chinese pre-trained language models. *arXiv preprint arXiv:2104.07204*.

Gina-Anne Levow. 2006. The third international Chinese language processing bakeoff: Word segmentation and named entity recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 108–117, Sydney, Australia. Association for Computational Linguistics.

Xiaonan Li, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020. Flat: Chinese ner using flat-lattice transformer. *arXiv preprint arXiv:2004.11795*.

Ruotian Ma, Minlong Peng, Qi Zhang, and Xuanjing Huang. 2019. Simplify the usage of lexicon in chinese ner. *arXiv preprint arXiv:1908.05969*.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. *arXiv preprint arXiv:1906.03158*.

Jianlin Su. 2020. Wobert: Word-based chinese bert model - zhuiyiai. Technical report.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, Weixin Liu, Zhihua Wu, Weibao Gong, Jianzhong Liang, Zhizhou Shang, Peng Sun, Wei Liu, Xuan Ouyang, Dianhai Yu, Hao Tian, Hua Wu, and Haifeng Wang. 2021a. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Zijun Sun, Xiaoya Li, Xiaofei Sun, Yuxian Meng, Xiang Ao, Qing He, Fei Wu, and Jiwei Li. 2021b. Chinesebert: Chinese pretraining enhanced by glyph and pinyin information. *arXiv preprint arXiv:2106.16038*.

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.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, et al. 2020. Clue: A chinese language understanding evaluation benchmark. *arXiv preprint arXiv:2004.05986*.

Haisong Zhang, Lemao Liu, Haiyun Jiang, Yangming Li, Enbo Zhao, Kun Xu, Linfeng Song, Suncong Zheng, Botong Zhou, Jianchen Zhu, Xiao Feng, Tao Chen, Tao Yang, Dong Yu, Feng Zhang, Zhanhui Kang, and Shuming Shi. 2020. Texsmart: A text understanding system for fine-grained ner and enhanced semantic analysis. *arXiv preprint arXiv:2012.15639*.

Xinsong Zhang and Hang Li. 2020. Ambert: A pre-trained language model with multi-grained tokenization. *arXiv preprint arXiv:2008.11869*.

Yue Zhang and Jie Yang. 2018. Chinese ner using lattice lstm. *arXiv preprint arXiv:1805.02023*.

Zexuan Zhong and Danqi Chen. 2020. A frustratingly easy approach for entity and relation extraction. *arXiv preprint arXiv:2010.12812*.