ChineseBERT: Chinese Pretraining Enhanced by Glyph and Pinyin Information

Zijun Sun♣, Xiaoya Li♣, Xiaofei Sun♣, Yuxian Meng♣
Xiang Ao♠, Qing He♠, Fei Wu♦ and Jiwei Li♦♣
♣Shannon.AI, ♦Zhejiang University
♠Key Lab of Intelligent Information Processing of Chinese Academy of Sciences
{zijun_sun, xiaoya_li, xiaofei_sun, yuxian_meng, jiwei_li}@shannonai.com
{aoxiang,heqing}@ict.ac.cn, wufei@zju.edu.cn

Abstract

Recent pretraining models in Chinese neglect two important aspects specific to the Chinese language: glyph and pinyin, which carry significant syntax and semantic information for language understanding. In this work, we propose ChineseBERT, which incorporates both the *glyph* and *pinyin* information of Chinese characters into language model pretraining. The glyph embedding is obtained based on different fonts of a Chinese character, being able to capture character semantics from the visual features, and the pinyin embedding characterizes the pronunciation of Chinese characters, which handles the highly prevalent heteronym phenomenon in Chinese (the same character has different pronunciations with different meanings). Pretrained on large-scale unlabeled Chinese corpus, the proposed ChineseBERT model yields significant performance boost over baseline models with fewer training steps. The proposed model achieves new SOTA performances on a wide range of Chinese NLP tasks, including machine reading comprehension, natural language inference, text classification, sentence pair matching, and competitive performances in named entity recognition and word segmentation.[1]

1 Introduction

Large-scale pretrained models have become a fundamental backbone for various natural language processing tasks such as natural language understanding (Liu et al., 2019b), text classification (Reimers and Gurevych, 2019; Chai et al., 2020) and question answering (Clark and Gardner, 2017; Lewis et al., 2020). Apart from English NLP tasks, pretrained models have also demonstrated their effectiveness for various Chinese NLP tasks (Sun et al., 2019, 2020; Cui et al., 2019a, 2020).

[1]The code and pretrained models are publicly available at `https://github.com/ShannonAI/ChineseBert`.

Since pretraining models are originally designed for English, two important aspects specific to the Chinese language are missing in current large-scale pretraining: glyph-based information and pinyin-based information. For the former, a key aspect that makes Chinese distinguishable from languages such as English, German, is that Chinese is a logographic language. The logographic of characters encodes semantic information. For example, "液(liquid)", "河(river)" and "湖(lake)" all have the radical "氵(water)", which indicates that they are all related to water in semantics. Intuitively, the rich semantics behind Chinese character glyphs should enhance the expressiveness of Chinese NLP models. This idea has motivated a variety of of work on learning and incorporating Chinese glyph information into neural models (Sun et al., 2014; Shi et al., 2015; Liu et al., 2017; Dai and Cai, 2017; Su and Lee, 2017; Meng et al., 2019), but not yet large-scale pretraining.

For the latter, *pinyin*, the Romanized sequence of a Chinese character representing its pronunciation(s), is crucial in modeling both semantic and syntax information that can not be captured by contextualized or glyph embeddings. This aspect is especially important considering the highly prevalent heteronym phenomenon in Chinese[2], where the same character have multiple pronunciations, each of which is associated with a specific meaning. Each pronunciation is associated with a specific pinyin expression. At the semantic level, for example, the Chinese character "乐" has two distinctly different pronunciations: "乐" can be pronounced as "yuè [yɛ^{51}]", which means "music", and "lè [lɤ^{51}]", which means "happy". On the syntax level, pronunciations help identify the part-of-speech of a character. For example, character "还" has two

[2]Among 7000 common characters in Chinese, there are about 700 characters that have multiple pronunciations, according to the Contemporary Chinese Dictionary.

pronunciations: “huán[xwan^{35}]” and “hái[xaɪ^{35}]”, with the former meaning the verb “return” and the latter meaning the adverb “also”. Different pronunciations of the same character cannot be distinguished by the glyph embedding since the logographic is the same, or the char-ID embedding, since they both point to the same character ID, but can be characterized by pinyin.

In this work, we propose ChineseBERT, a model that incorporates the glyph and pinyin information of Chinese characters into the process of large-scale pretraining. The glyph embedding is based on different fonts of a Chinese character, being able to capture character semantics from the visual surface character forms. The pinyin embedding models different semantic meanings that share the same character form and thus bypasses the limitation of interwound morphemes behind a single character. For a Chinese character, the glyph embedding, the pinyin embedding and the character embedding are combined to form a fusion embedding, which models the distinctive semantic property of that character.

With less training data and fewer training epochs, ChineseBERT achieves significant performance boost over baselines across a wide range of Chinese NLP tasks. It achieves new SOTA performances on a wide range of Chinese NLP tasks， including machine reading comprehension, natural language inference, text classification, sentence pair matching, and results comparable to SOTA performances in named entity recognition and word segmentation.

2 Related Work

2.1 Large-Scale Pretraining in NLP

Recent years has witnessed substantial work on large-scale pretraining in NLP. BERT (Devlin et al., 2018), which is built on top of the Transformer architecture (Vaswani et al., 2017), is pretrained on large-scale unlabeled text corpus in the manner of Masked Language Model (MLM) and Next Sentence Prediction (NSP). Following this trend, considerable progress has been made by modifying the masking strategy (Yang et al., 2019; Joshi et al., 2020), pretraining tasks (Liu et al., 2019a; Clark et al., 2020) or model backbones (Lan et al., 2020; Lample et al., 2019; Choromanski et al., 2020). Specifically, RoBERTa (Liu et al., 2019b) proposed to remove the NSP pretraining task since it has been proved to offer no benefits for improving downstream performances. The GPT series (Radford et al., 2019; Brown et al., 2020) and other BERT variants (Lewis et al., 2019; Song et al., 2019; Lample and Conneau, 2019; Dong et al., 2019; Bao et al., 2020; Zhu et al., 2020) adapted the paradigm of large-scale unsupervised pretraining to text generation tasks such as machine translation, text summarization and dialog generation, so that generative models can enjoy the benefit of large-scale pretraining.

Unlike the English language, Chinese has its particular characteristics in terms of syntax, lexicon and pronunciation. Hence, pretraining Chinese models should fit the Chinese features correspondingly. Li et al. (2019b) proposed to use Chinese character as the basic unit instead of word or subword that is used in English (Wu et al., 2016; Sennrich et al., 2016). ERNIE (Sun et al., 2019, 2020) applied three types of masking strategies – char-level masking, phrase-level masking and entity-level masking – to enhance the ability of capturing multi-granularity semantics. Cui et al. (2019a, 2020) pretrained models using the Whole Word Masking strategy, where all characters within a Chinese word are masked altogether. In this way, the model is learning to address a more challenging task as opposed to predicting word components. More recently, Zhang et al. (2020) developed the largest Chinese pretrained language model to date – CPM. It is pretrained on 100GB Chinese data and has 2.6B parameters comparable to “GPT3 2.7B” (Brown et al., 2020). Xu et al. (2020) released the first large-scale Chinese Language Understanding Evaluation benchmark CLUE, facilitating researches in large-scale Chinese pretraining.

2.2 Learning Glyph Information

Learning glyph information from surface Chinese character forms has gained attractions since the prevalence of deep neural networks. Inspired by word embeddings (Mikolov et al., 2013b,a), Sun et al. (2014); Shi et al. (2015); Li et al. (2015); Yin et al. (2016) used indexed radical embeddings to capture character semantics, improving model performances on a wide range of Chinese NLP tasks. Another way of incorporating glyph information is to view characters in the form of image, by which glyph information can be naturally learned through image modeling. However, early work on learning visual features is not smooth. Liu et al. (2017); Shao et al. (2017); Zhang and LeCun (2017); Dai

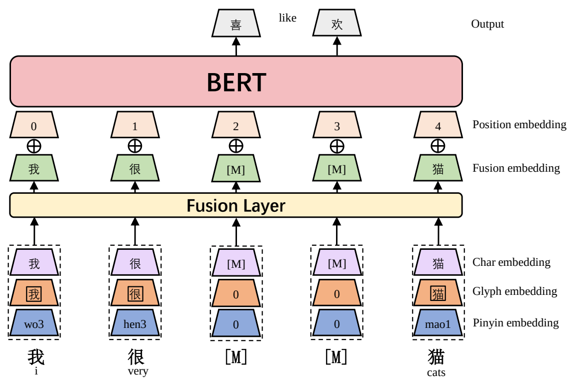

Figure 1: An overview of ChineseBERT. The fusion layer consumes three D-dimensional embeddings – char embedding, glyph embedding and pinyin embedding. The three embeddings are first concatenated, and then mapped to a D-dimensional embedding through a fully connected layer to form the fusion embedding.

and Cai (2017) used CNNs to extract glyph features from character images but did not achieve consistent performance boost over all tasks. Su and Lee (2017); Tao et al. (2019) obtained positive results on the word analogy and word similarity tasks but they did not further evaluate the learned glyph embeddings on more tasks. Meng et al. (2019) applied glyph embeddings to a broad array of Chinese tasks. They designed a specific CNN structure for character feature extraction and used image classification as an auxiliary objective to regularize the influence of a limited number of images. Song and Sehanobish (2020); Xuan et al. (2020) extended the idea of Meng et al. (2019) to the task of named entity recognition (NER), significantly improving performances against vanilla BERT models.

3 Model

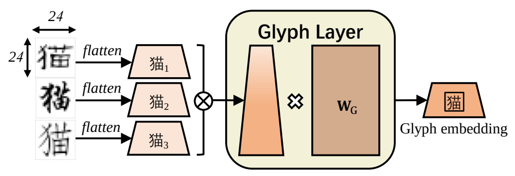

Figure 2: An overview of inducing the *glyph* embedding. ⊗ denotes vector concatenation. For each Chinese character, we use three types of fonts – *FangSong*, *XingKai* and *LiShu*, each of which is a 24×24 image with pixel value ranging $0 \sim 255$. Images are concatenated into a tensor of size $24 \times 24 \times 3$. The tensor is flattened and passed to an FC to obtain the glyph embedding.

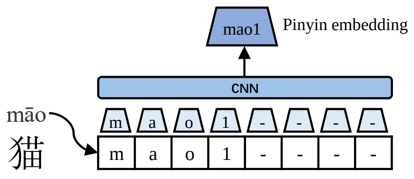

Figure 3: An overview of inducing the *pinyin* embedding. For any Chinese character, e.g. 猫(cat) in this case, a CNN with width 2 is applied to the sequence of Romanized pinyin letters, followed by max-pooling to derive the final pinyin embedding.

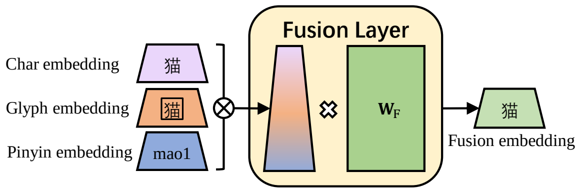

Figure 4: An overview of the fusion layer. ⊗ denotes vector concatenation, and × is vector-matrix multiplication. We concatenate the char embedding, the glyph embedding and the pinyin embedding, and use an FC layer with a learnable matrix $\mathbf{W}_F$ to induce the fusion embedding.

3.1 Overview

Figure 1 shows an overview of the proposed ChineseBERT model. For each Chinese character, its char embedding, glyph embedding and pinyin embedding are first concatenated, and then mapped to a D-dimensional embedding through a fully connected layer to form the fusion embedding. The fusion embedding is then added with the position embedding, which is fed as input to the BERT model Since we do not use the NSP pretraining task, we omit the segment embedding. We use both Whole Word Masking (WWM) (Cui et al., 2019a) and Char Masking (CM) for pretraining (See Section 4.2 for details).

3.2 Input

The input to the model is the addition of the learnable absolute positional embedding and the fusion embedding, where the fusion embedding is based on the char embedding, the glyph embedding and the pinyin embedding of the corresponding character. The char embedding performs in a way analogous to the token embedding used in BERT but at the character granularity. Below we respectively describe how to induce the glyph embedding, the pinyin embedding and the fusion embedding.

Glyph Embedding We follow Meng et al. (2019) to use three types of Chinese fonts – FangSong, XingKai and LiShu, each of which is instantiated as a 24×24 image with floating point pixels ranging from 0 to 255. The $24 \times 24 \times 3$ vector is first flattened to a 2,352 vector. The flattened vector is fed to an FC layer to obtain the output glyph vector.

Pinyin Embedding The *pinyin* embedding for each character is used to decouple different semantic meanings belonging to the same character form, as shown in Figure 3. We use the opensourced pypinyin package[3] to generate pinyin sequences for its constituent characters. pypinyin is a system that combines machine learning models with dictionary-based rules to infer the pinyin for characters given contexts. Pinyin for a Chinese character is a sequence of Romanian characters, with one of four diacritics denoting tones. We use special tokens to denote tones, which are appended to the end of the Romanian character sequence. We apply a CNN model with width 2 on the pinyin sequence, followed by max-pooling to derive the resulting pinyin embedding. This makes output dimensionality immune to the length of the input pinyin sequence. The length of the input pinyin sequence is fixed at 8, with the remaining slots filled with a special letter "-" when the actual length of the pinyin sequence does not reach 8.

Fusion Embedding Once we have the char embedding, the glyph embedding and the pinyin embedding for a character, we concatenate them to form a $3D$-dimensional vector. The fusion layers maps the $3D$-dimensional vector to D-dimensional through a fully connected layer. The fusion embedding is added with position embedding, and output to the BERT layer. An illustration is shown in Figure 4.

3.3 Output

The output is the corresponding contextualized representation for each input Chinese character (Devlin et al., 2018).

4 Pretraining Setup

4.1 Data

We collected our pretraining data from CommonCrawl[4]. After pre-processing (such as removing the data with too much English text and filtering the `html` tagger), about 10% high-quality data is maintained for pretraining, containing 4B Chinese characters in total. We use the LTP toolkit[5] (Che et al., 2010) to identify the boundary of Chinese words for whole word masking.

4.2 Masking Strategies

We use two masking strategies – Whole Word Masking (WWM) and Char Masking (CM) for ChineseBERT. Li et al. (2019b) suggested that using Chinese characters as the basic input unit can alleviate the out-of-vocabulary issue in the Chinese language. We thus adopt the method of masking random characters in the given context, denoted by Char Masking. On the other hand, a large number of words in Chinese consist of multiple characters, for which the CM strategy may be too easy for the model to predict. For example, for the input context "我喜欢逛紫禁[M] (i like going to The Forbidden [M])", the model can easily predict that the masked character is "城(City)". Hence, we follow Cui et al. (2019a) to use WWM, a strategy to mask out all characters within a selected word, mitigating the easy-predicting shortcoming of the CM strategy. Note that for both WWM and CM, the basic input unit is Chinese characters. The main difference between WWM and CM lies in how they mask characters and how the model predicts masked characters.

4.3 Pretraining Details

Different from Cui et al. (2019a) who pretrained their model based on the official pretrained Chinese BERT model, we train the ChineseBERT model from scratch. To enforce the model to learn both long-term and short-term dependencies, we propose to alternate pretraining between *packed input* and *single input*, where the packed input is the concatenation of multiple sentences with a maximum length 512, and the single input is a single sentence. We feed the packed input with probability of 0.9 and the single input with probability of 0.1. We apply Whole Word Masking 90% of the time and Char Masking 10% of the time. The masking probability for each word/char is 15%. If the i-th word/char is chosen, we mask it 80% of the time, replace it with a random word/char 10% of the time and maintain it 10% of the time. We also use the dynamic masking strategy to avoid duplicate training instances (Liu et al., 2019b). We use

[3]https://pypi.org/project/pypinyin/
[4]https://commoncrawl.org/
[5]http://ltp.ai/

	ERNIE	**BERT-wwm**	**MacBERT**	**ChineseBERT**
Data Source	Heterogeneous	Wikipedia	Heterogeneous	CommonCrawl
Vocab Size	18K	21K	21K	21K
Input Unit	Char	Char	Char	Char
Masking	T/P/E	WWM	WWM/N	WWM/CM
Task	MLM/NSP	MLM	MAC/SOP	MLM
Training Steps	-	2M	1M	1M
Init Checkpoint		BERT	BERT	random
# Token	–	0.4B	5.4B	5B

Table 1: Comparison of data statistics between ERNIE (Sun et al., 2019), BERT-wwm (Cui et al., 2019a), MacBERT (Cui et al., 2020) and our proposed ChineseBERT. T: Token, P: Phrase, E: Entity, WWM: Whole Word Masking, N: N-gram, CM: Char Masking, MLM: Masked Language Model, NSP: Next Sentence Prediction, MAC: MLM-As-Correlation. SOP: Sentence Order Prediction.

two model setups: `base` and `large`, respectively consisting of 12/24 Transformer layers, with input dimensionality of 768/1,024 and 12/16 heads per layer. This makes our models comparable to other BERT-style models in terms of model size. Upon the submission of the paper, we have trained the `base` model 500K steps with a maximum learning rate 1e-4, warmup of 20K steps and a batch size of 3.2k, and the `large` model 280K steps with a maximum learning rate 3e-4, warmup of 90K steps and a batch size of 8k. After pretraining, the model can be directly used to be finetuned on downstream tasks in the same way as BERT (Devlin et al., 2018).

5 Experiments

We conduct extensive experiments on a variety of Chinese NLP tasks. Models are separately fine-tuned on task-specific datasets for evaluation. Concretely, we use the following tasks:

- Machine Reading Comprehension (MRC)
- Natural Language Inference (NLI)
- Text Classification (TC)
- Sentence Pair Matching (SPM)
- Named Entity Recognition (NER)
- Chinese Word Segmentation (CWS).

We compare ChineseBERT to current state-of-the-art ERNIE (Sun et al., 2019, 2020), BERT-wwm (Cui et al., 2019a) and MacBERT (Cui et al., 2020) models. ERNIE adopts various masking strategies including token-level, phrase-level and entity-level masking to pretrain BERT on large-scale heterogeneous data. BERT-wwm/RoBERTa-wwm continues pretraining on top of official pretrained Chinese BERT/RoBERTa models with the Whole Word Masking pretraining strategy. Unless otherwise specified, we use BERT/RoBERTa to represent BERT-wwm/RoBERTa-wwm and omit "wwm". MacBERT improves upon RoBERTa by using the MLM-As-Correlation (MAC) pretraining strategy as well as the sentence-order prediction (SOP) task. It is worth noting that BERT and BERT-wwm do not have the large version available online, and we thus omit the corresponding performances.

A comparison of these models is shown in Table 1. It is **worth noting** that training steps of the proposed model significantly smaller than baseline models. Different from BERT-wwm and MacBERT which are initialized with pretrained BERT, the proposed model is initialized from scratch. Due to the additional consideration of glyph and pinyin, the proposed cannot be directly initialized using a vanilla BERT model, as the model structures are different. Even initialized from scratch, the proposed model is trained fewer steps than the steps in retraining BERT-wwm and MacBERT after BERT initialization.

5.1 Machine Reading Comprehension

Machine reading comprehension tests the model's ability of answering the questions based on the given contexts. We use two datasets for this task: CMRC 2018 (Cui et al., 2019b) and CJRC (Duan et al., 2019) . CMRC is a span-extraction style dataset while CJRC additionally has yes/no questions and no-answer questions. CMRC 2018 and CJRC respectively contain 10K/3.2K/4.9K and 39K/6K/6K data instances for training/dev/test. Test results for CMRC 2018 are evaluated from the CLUE leaderboard.[6] Note that the CJRC dataset is different from the one used in Cui et al. (2019a) as Cui et al. (2019a) did not release their train/dev/test split. We thus run the released models on the CJRC dataset used in this work for comparison.

Results are shown in Table 2 and Table 3. As we can see, ChineseBERT yields significant performance boost on both datasets, and the improvement of EM is larger than that of F1 on the CJRC dataset, which indicates that ChineseBERT is better at detecting exact answer spans.

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

Model	*CMRC* Dev	Test
	Base	
ERNIE	66.89	74.70
BERT	66.77	71.60
BERT°	66.96	73.95
RoBERTa°	67.89	75.20
MacBERT	–	–
ChineseBERT	**67.95**	**75.35**
	Large	
RoBERTa°	70.59	77.95
MacBERT	–	–
ChineseBERT	**70.70**	**78.05**

Table 2: Performances of different models on CMRC. EM is reported for comparison. ○ represents models pretrained on extended data.

	CJRC			
	Dev		Test	
Model	EM	F1	EM	F1
	Base			
BERT	59.8	73.0	60.2	73.0
BERT°	60.8	74.0	61.4	73.9
RoBERTa°	62.9	76.6	63.8	76.6
ChineseBERT	**65.2**	**77.8**	**66.2**	**77.9**
	Large			
RoBERTa°	65.6	77.5	66.4	77.6
ChineseBERT	**66.5**	**77.9**	**67.0**	**78.3**

Table 3: Performances of different models on the MRC dataset CJRC. We report results for baseline models based on their released models. ○ represents models pretrained on extended data.

5.2 Natural Language Inference (NLI)

The goal of NLI is to determine the entailment relationship between a hypothesis and a premise. We use the Cross-lingual Natural Language Inference (XNLI) dataset (Conneau et al., 2018) for evaluation. The corpus is a crowd-sourced collection of 5K test and 2.5K dev pairs for the MultiNLI corpus. Each sentence pair is annotated with the "entailment", "neutral" or "contradiction" label. We use the official machine translated Chinese data for training.[7]

Results are present in Table 4, which shows that ChineseBERT is able to achieve the best performances for both `base` and `large` setups.

5.3 Text Classification (TC)

In text classification the model is required to categorize a piece of text into one of the specified classes. We follow Cui et al. (2019a) to use THUCNews (Li and Sun, 2007) and ChnSentiCorp[8] for this task. THUCNews is a subset of THUCTC [9], with 50K/5K/10K data points respectively for training/dev/test. Data is evenly distributed in 10 domains including sports, finance, etc.[10] ChnSentiCorp is a binary sentiment classification dataset containing 9.6K/1.2K/1.2K data points respectively for training/dev/test. The two datasets are relatively simple with vanilla BERT achieving an accuracy of above 95%. Hence, apart from THUCNews and ChnSentiCorp, we also use TNEWS, a more difficult dataset that is included in the CLUE benchmark (Xu et al., 2020).[11] TNEWS is a 15-class short news text classification dataset with 53K/10K/10K data points for training/dev/test.

Results are shown in Table 5. On ChunSentiCorp and THUCNews, the improvement from ChineseBERT is marginal as baselines have already achieved quite high results on these two datasets. On the TNEWS dataset, ChineseBERT outperforms all other models. We can see that the ERNIE model only performs slightly worse than ChineseBERT. This is because ERNIE is trained on additional web data, which is beneficial to model web news text that covers a wide range of domains.

Model	*XNLI* Dev	Test
	Base	
ERNIE	79.7	78.6
BERT	79.0	78.2
BERT°	79.4	78.7
RoBERTa°	80.0	78.8
MacBERT	80.3	79.3
ChineseBERT	**80.5**	**79.6**
	Large	
RoBERTa°	82.1	81.2
MacBERT	82.4	81.3
ChineseBERT	**82.7**	**81.6**

Table 4: Performances of different models on XNLI. Accuracy is reported for comparison. ○ represents models pretrained on extended data.

5.4 Sentence Pair Matching (SPM)

For SPM, the model is asked to determine whether a given sentence pair expresses the same semantics. We use the LCQMC (Liu et al., 2018) and BQ Corpus (Chen et al., 2018) datasets for evaluation.

[7] `https://github.com/facebookresearch/XNLI`

[8] `https://github.com/pengming617/bert_classification/tree/master/data`

[9] `http://thuctc.thunlp.org/`

[10] `https://github.com/gaussic/text-classification-cnn-rnn`

[11] `https://github.com/CLUEbenchmark/CLUE`

Model	*ChnSentiCorp* Dev	*ChnSentiCorp* Test	*THUCNews* Dev	*THUCNews* Test	*TNEWS* Dev	*TNEWS* Test
			Base			
ERNIE	95.4	95.5	97.6	97.5	58.24	58.33
BERT	95.1	95.4	98.0	97.8	56.09	56.58
BERT°	95.4	95.3	97.7	97.7	56.77	56.86
RoBERTa°	95.0	95.6	**98.3**	97.8	57.51	56.94
MacBERT	95.2	95.6	98.2	97.7	–	–
ChineseBERT	**95.6**	**95.7**	98.1	**97.9**	**58.64**	**58.95**
			Large			
RoBERTa°	**95.8**	95.8	**98.3**	97.8	58.32	58.61
MacBERT	95.7	**95.9**	98.1	**97.9**	–	–
ChineseBERT	**95.8**	**95.9**	**98.3**	**97.9**	**59.06**	**59.47**

Table 5: Performances of different models on TC datasets ChnSentiCorp, THUCNews and TNEWS. The results of TNEWS are taken from the CLUE paper (Xu et al., 2020). Accuracy is reported for comparison. ◦ represents models pretrained on extended data.

Model	*LCQMC* Dev	*LCQMC* Test	*BQ Corpus* Dev	*BQ Corpus* Test
		Base		
ERNIE	89.8	87.2	86.3	85.0
BERT	89.4	87.0	86.1	85.2
BERT°	89.6	87.1	**86.4**	**85.3**
RoBERTa°	89.0	86.4	86.0	85.0
MacBERT	89.5	87.0	86.0	85.2
ChineseBERT	**89.8**	**87.4**	**86.4**	85.2
		Large		
RoBERTa°	90.4	87.0	86.3	85.8
MacBERT	**90.6**	87.6	86.2	85.6
ChineseBERT	90.5	**87.8**	**86.5**	**86.0**

Table 6: Performances of different models on SPM datasets LCQMC and BQ Corpus. We report accuracy for comparison. ◦ represents models pretrained on extended data.

LCQMC is a large-scale Chinese question matching corpus for judging whether two given questions have the same intent, with 23.9K/8.8K/12.5K sentence pairs for training/dev/test. BQ Corpus is another large-scale Chinese dataset containing 100K/10K/10K sentence pairs for training/dev/test. Results are shown in Table 6. We can see that ChineseBERT generally outperforms MacBERT on LCQMC but slightly underperforms BERT-wwm. We hypothesis this is because the domain of BQ Corpus more fits the pretraining data of BERT-wwm than that of ChineseBERT.

5.5 Named Entity Recognition (NER)

For NER tasks (Chiu and Nichols, 2016; Lample et al., 2016; Li et al., 2019a), the model is asked to identify named entities within a piece of text, which is formalized as a sequence labeling task. We use OntoNotes 4.0 (Weischedel et al., 2011) and Weibo (Peng and Dredze, 2015) for this task.

Model	*OntoNotes 4.0* P	*OntoNotes 4.0* R	*OntoNotes 4.0* F	*Weibo* P	*Weibo* R	*Weibo* F
			Base			
BERT	79.69	82.09	80.87	67.12	66.88	67.33
RoBERTa°	**80.43**	80.30	80.37	**68.49**	67.81	68.15
ChineseBERT	80.03	**83.33**	**81.65**	68.27	**69.78**	**69.02**
			Large			
RoBERTa°	80.72	82.07	81.39	66.74	70.02	68.35
ChineseBERT	**80.77**	**83.65**	**82.18**	**68.75**	**72.97**	**70.80**

Table 7: Performances of different models on NER datasets OntoNotes 4.0 and Weibo. Results of precision (P), recall (R) and F1 (F) on test sets are reported for comparison.

Model	*MSRA* F1	*MSRA* Acc	*PKU* F1	*PKU* Acc
		Base		
BERT°	98.42	99.04	96.82	97.70
RoBERTa°	98.46	99.10	96.88	97.72
ChineseBERT	**98.60**	**99.14**	**97.02**	**97.81**
		Large		
RoBERTa°	98.49	99.13	96.95	97.80
ChineseBERT	**98.67**	**99.26**	**97.16**	**98.01**

Table 8: Performances of different models on CWS datasets MSRA and PKU. We report F1 and accuracy (Acc) for comparison. ◦ represents models pretrained on extended data.

We use OntoNotes 4.0 and Weibo NER for this task. OntoNotes has 18 named entity types and Weibo has 4 named entity types. OntoNotes and Weibo respectively contain 15K/4K/4K and 1,350/270/270 instances for training/dev/test. Results are shown in Table 7. As we can see, ChineseBERT significantly outperforms BERT and RoBERTa in terms of F1. In spite of a slight loss on precision for the base version, the gains on recall are particularly high, leading to a final performance boost on F1.

5.6 Chinese Word Segmentation

The task divides text into words and is formalized as a character-based sequence labelling task. We use the PKU and MSRA datasets for Chinese word segmentation. PKU consists of 19K/2K sentences for training and test, and MSRA consists of 87k/4k sentences for training and test. Output character embedding is fed to the softmax function for final predictions. Results are shown in Table 8, where we can see that ChineseBERT is able to outperform BERT-wwm and RoBERTa-wwm on both datasets for both metrics.

6 Ablation Studies

In this section, we conduct ablation studies to understand the behaviors of ChineseBERT. We

	OntoNotes 4.0		
Model	**Precision**	**Recall**	**F1**
RoBERTa$^\circ$	80.43	80.30	80.37
ChineseBERT	80.03	83.33	81.65
– Glyph	77.67	82.75	80.13 (-1.52)
– Pinyin	77.54	83.65	80.48 (-1.17)
– Glyph – Pinyin	78.22	81.37	79.76 (-1.89)

Table 9: Performances for different models without considering glyph or pinyin information.

use the Chinese named entity recognition dataset OntoNotes 4.0 for analysis and all models are based on the `base` version.

6.1 The Effect of Glyph Embeddings and Pinyin Embeddings

We would like to explore the effects of glyph embeddings and pinyin embeddings. For fair comparison, we pretrained different models on the same dataset, with the same number of training steps, and with the same model size. Setups include "-glyph", where glyph embeddings are not considered and we only consider pinyin and char-ID embeddings; "-pinyin", where pinyin embeddings are not considered and we only consider glyph and char-ID embeddings; "-glyph-pinyin", where only char-ID embeddings are considered, and the model degenerates to RoBERTa. We finetune different models on the OntoNotes dataset of the NER dataset for comparison.

Results are shown in Table 9. As can be seen, either removing glyph embeddings or pinyin embeddings results in performance degradation, and removing both has the greatest negative impact on the F1 value, which is a drop of about 2 points. This validates the importance of both pinyin and glyph embeddings for modeling Chinese semantics. The reason why "-glyph-pinyin" performs worse than RoBERTa is that the model we use here is trained on a smaller size of data with smaller number of training steps.

6.2 The Effect of Training Data Size

We hypothesize glyph and pinyin embeddings also serve as strong regularization over text semantics, which means that the proposed ChineseBERT model is able to perform better with less training data. We randomly sample 10%∼90% of the training data while maintaining the ratio of samples with entities w.r.t. samples without entities. We perform each experiment five times and report the average F1 value on the test set. Figure 5 shows the results. As can be seen, ChineseBERT performs better across all setups. With less than 30% of the training data, the improvement of ChineseBERT is slight, but with over 30% training data, the performance improvement is greater. This is because ChineseBERT still requires sufficient training data to fully train the glyph and pinyin embeddings, and insufficient training data would lead to inadequate training.

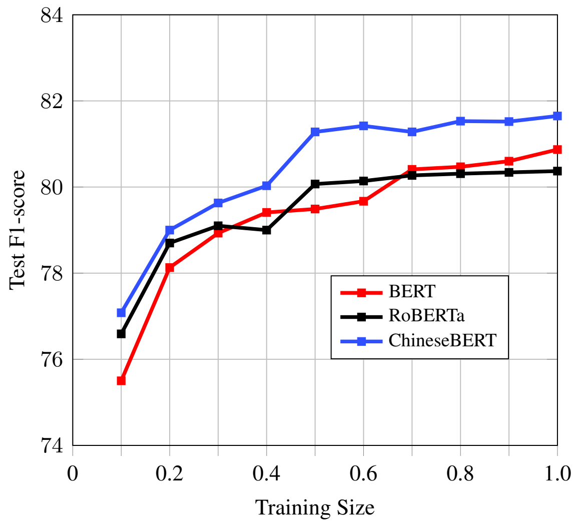

Figure 5: Performances when varying the training size.

7 Conclusion

In this paper, we introduce ChineseBERT, a large-scale pretraining Chinese NLP model. It leverages the glyph and pinyin information of Chinese characters to enhance the model's ability of capturing context semantics from surface character forms and disambiguating polyphonic characters in Chinese. The proposed ChineseBERT model achieves significant performance boost across a wide range of Chinese NLP tasks. The proposed ChineseBERT performs better than vanilla pretrained models with less training data, indicating that the introduced glyph embeddings and pinyin embeddings serve as a strong regularizer for semantic modeling in Chinese. Future work involves training a large size version of ChineseBERT.

Acknowledgement

This work is supported by National Key R&D Program of China (2020AAA0105200) and Beijing Academy of Artificial Intelligence (BAAI).

References

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Songhao Piao, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2020. Unilmv2: Pseudo-masked language models for unified language model pre-training.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Duo Chai, Wei Wu, Qinghong Han, Fei Wu, and Jiwei Li. 2020. Description based text classification with reinforcement learning.

Wanxiang Che, Zhenghua Li, and Ting Liu. 2010. LTP: A Chinese language technology platform. In *Coling 2010: Demonstrations*, pages 13–16, Beijing, China. Coling 2010 Organizing Committee.

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, Brussels, Belgium. Association for Computational Linguistics.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. 2020. Rethinking attention with performers. *CoRR*.

Christopher Clark and Matt Gardner. 2017. Simple and effective multi-paragraph reading comprehension. *arXiv preprint arXiv:1710.10723*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Alexis Conneau, Guillaume Lample, Ruty Rinott, Adina Williams, Samuel R Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. *arXiv preprint arXiv:1809.05053*.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for chinese natural language processing. *arXiv preprint arXiv:2004.13922*.

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.

Falcon Z Dai and Zheng Cai. 2017. Glyph-aware embedding of chinese characters. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP, Copenhagen, Denmark, September 7, 2017*, pages 64–69.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, volume 32, pages 13063–13075. Curran Associates, Inc.

Xingyi Duan, Baoxin Wang, Ziyue Wang, Wentao Ma, Yiming Cui, Dayong Wu, Shijin Wang, Ting Liu, Tianxiang Huo, Zhen Hu, and et al. 2019. Cjrc: A reliable human-annotated benchmark dataset for chinese judicial reading comprehension. *Chinese Computational Linguistics*, page 439–451.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *Advances in Neural Information Processing Systems (NeurIPS)*.

Guillaume Lample, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2019. Large memory layers with product keys. *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Jingyang Li and Maosong Sun. 2007. Scalable term selection for text categorization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 774–782.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2019a. A unified mrc framework for named entity recognition. *arXiv preprint arXiv:1910.11476*.

Xiaoya Li, Yuxian Meng, Xiaofei Sun, Qinghong Han, Arianna Yuan, and Jiwei Li. 2019b. Is word segmentation necessary for deep learning of Chinese representations? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3242–3252, Florence, Italy. Association for Computational Linguistics.

Yanran Li, Wenjie Li, Fei Sun, and Sujian Li. 2015. Component-enhanced chinese character embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 829–834.

Frederick Liu, Han Lu, Chieh Lo, and Graham Neubig. 2017. Learning character-level compositionality with visual features. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 2059–2068.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, Florence, Italy. Association for Computational Linguistics.

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.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yuxian Meng, Wei Wu, Fei Wang, Xiaoya Li, Ping Nie, Fan Yin, Muyu Li, Qinghong Han, Xiaofei Sun, and Jiwei Li. 2019. Glyce: Glyph-vectors for chinese character representations. In *Advances in Neural Information Processing Systems*, volume 32, pages 2746–2757. Curran Associates, Inc.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26:3111–3119.

Nanyun Peng and Mark Dredze. 2015. Named entity recognition for chinese social media with jointly trained embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 548–554.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

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.

Yan Shao, Christian Hardmeier, Jörg Tiedemann, and Joakim Nivre. 2017. Character-based joint segmentation and pos tagging for chinese using bidirectional rnn-crf. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP 2017, Taipei, Taiwan, November 27 - December 1, 2017 - Volume 1: Long Papers*, pages 173–183.

Xinlei Shi, Junjie Zhai, Xudong Yang, Zehua Xie, and Chao Liu. 2015. Radical embedding: Delving deeper to Chinese radicals. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International*

Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 594–598, Beijing, China. Association for Computational Linguistics.

Chan Hee Song and Arijit Sehanobish. 2020. Using chinese glyphs for named entity recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(10):13921–13922.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning*, pages 5926–5936.

Tzu-Ray Su and Hung-Yi Lee. 2017. Learning chinese word representations from glyphs of characters. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 264–273.

Yaming Sun, Lei Lin, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2014. Radical-enhanced chinese character embedding. In *International Conference on Neural Information Processing*, pages 279–286. Springer.

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*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8968–8975.

Hanqing Tao, Shiwei Tong, Tong Xu, Qi Liu, and Enhong Chen. 2019. Chinese embedding via stroke and glyph information: A dual-channel view.

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, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, Robert Belvin, et al. 2011. Ontonotes release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*.

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, Łukasz 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.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, Yin Tian, Qianqian Dong, Weitang Liu, Bo Shi, Yiming Cui, Junyi Li, Jun Zeng, Rongzhao Wang, Weijian Xie, Yanting Li, Yina Patterson, Zuoyu Tian, Yiwen Zhang, He Zhou, Shaoweihua Liu, Zhe Zhao, Qipeng Zhao, Cong Yue, Xinrui Zhang, Zhengliang Yang, Kyle Richardson, and Zhenzhong Lan. 2020. CLUE: A Chinese language understanding evaluation benchmark. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4762–4772, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zhenyu Xuan, Rui Bao, and Shengyi Jiang. 2020. Fgn: Fusion glyph network for chinese named entity recognition.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

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.

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*.

Zhengyan Zhang, Xu Han, Hao Zhou, Pei Ke, Yuxian Gu, Deming Ye, Yujia Qin, Yusheng Su, Haozhe Ji, Jian Guan, Fanchao Qi, Xiaozhi Wang, Yanan Zheng, Guoyang Zeng, Huanqi Cao, Shengqi Chen, Daixuan Li, Zhenbo Sun, Zhiyuan Liu, Minlie Huang, Wentao Han, Jie Tang, Juanzi Li, Xiaoyan Zhu, and Maosong Sun. 2020. Cpm: A large-scale generative chinese pre-trained language model.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tieyan Liu. 2020. Incorporating bert into neural machine translation. In *International Conference on Learning Representations*.