NEZHA: NEURAL CONTEXTUALIZED REPRESENTATION FOR CHINESE LANGUAGE UNDERSTANDING

TECHNICAL REPORT

Junqiu Wei, Xiaozhe Ren, Xiaoguang Li, Wenyong Huang, Yi Liao,
Yasheng Wang, Jiashu Lin*, Xin Jiang, Xiao Chen, Qun Liu
Noah's Ark Lab, *HiSilicon, Huawei Technologies
{wei.junqiu1, renxiaozhe, lixiaoguang11, wenyong.huang, liao.yi,
wangyasheng, linjiashu, jiang.xin, chen.xiao2, qun.liu}@huawei.com

September 6, 2019

ABSTRACT

The pre-trained language models have achieved great successes in various natural language understanding (NLU) tasks due to its capacity to capture the deep contextualized information in text by pre-training on large-scale corpora. In this technical report, we present our practice of pre-training language models named NEZHA (NEural contextualiZed representation for CHinese lAnguage understanding) on Chinese corpora and finetuning for the Chinese NLU tasks. The current version of NEZHA is based on BERT [1] with a collection of proven improvements, which include *Functional Relative Positional Encoding* as an effective positional encoding scheme, *Whole Word Masking* strategy, *Mixed Precision Training* and the *LAMB Optimizer* in training the models. The experimental results show that NEZHA achieves the state-of-the-art performances when finetuned on several representative Chinese tasks, including named entity recognition (People's Daily NER), sentence matching (LCQMC), Chinese sentiment classification (ChnSenti) and natural language inference (XNLI).

***K*eywords** Pre-trained Language Models · NEZHA · Chinese Language Understanding

1 Introduction

Pre-trained language models such as ELMo [2], BERT [1], ERNIE-Baidu [3, 4], ERNIE-Tsinghua [5], XLNet [6], RoBERTa [7] and MegatronLM[1] have demonstrated remarkable successes in modeling contextualized word representations by utilizing the massive amount of training text. As a fundamental technique in natural language processing (NLP), the language models pre-trained on text could be easily transferred to learn downstream NLP tasks with fine-tuning, which achieve the state-of-the-art performances on many tasks including sentiment analysis, machine reading comprehension, sentence matching, named entity recognition and natural language inference.

The existing pre-trained language models are mostly learned from English corpora (e.g., BooksCorpus and English Wikipedia). There are several attempts to train the models specifically for the Chinese language, including Google's BERT [1] for Chinese, ERNIE-Baidu [3, 4] and BERT-WWM [8]. All of the models are based on Transformer [9] and trained on two unsupervised tasks: *Masked Language Model (MLM)* and *Next Sentence Prediction (NSP)*. In the MLM task, the model learns to recover the masked words in the training sentences. In the NSP task, it tries to predict whether one sentence is the next sentence of the other. One of the main differences among the Chinese models lies their word masking strategy in the MLM task. Google's BERT masks each Chinese character or WordPiece token [10] independently. ERNIE-Baidu further makes the MLM task more challenging by masking the entities or phrases in a sentence as a whole, where each entity or phrase may contain multiple characters or tokens. BERT-WWM takes a similar strategy called *Whole Word Masking (WWM)*, which enforces that all the tokens belonging to a Chinese word should be masked together. Besides, in the most recently published ERNIE-Baidu 2.0 [4], additional pre-training tasks such as *Token-Document Relation Prediction* and *Sentence Reordering*, are also incorporated.

[1] https://nv-adlr.github.io/MegatronLM

In this technical report, we present our practice of pre-training language models NEZHA (NEural contextualiZed representation for CHinese lAnguage understanding), which is currently based on BERT and trained on Chinese text. Specifically, we employ a technique called *Functional Relative Positional Encoding* in our model. In the vanilla Transformer as well as the BERT model, the positional encoding of each word in the sequence is a vector with its absolute position information encoded. The positional encodings are added to the word embeddings as the inputs to the Transformer. There are two typical ways to determine the positional encodings. One is the *functional* positional encoding, where the positional encodings are determined by some pre-defined functions (e.g., sinusoidal functions in [9]). The other is the *parametric* positional encodings, which are part of the model parameters and learned as in [1]. [11] proposes a parametric *relative* positional encoding, where the relative position information is incorporated in the self-attention layers of Transformer. Later, Transformer-XL [12] and XLNet [6] propose using a sinusoid encoding matrix and two trainable bias terms to represent the relative positions. In this technical report, we employ a functional relative positional encoding scheme, which encodes the relative positions in self-attention by pre-defined functions without any trainable parameter. Our empirical study shows that it is an effective positional encoding scheme for the pre-trained language models, and it makes consistent gains in our experiments. Besides, we employed three techniques shown to be effective in the pre-training of the BERT model, which are *Whole Word Masking* [8], *Mixed Precision Training* [13] and the *LAMB Optimizer* [14], in training NEZHA.

The contribution of this technical report is that we systematically study the problem of pre-training language models on large-scale Chinese corpora, evaluate the models on several Chinese NLU tasks, and assess the effectiveness of training factors including positional encoding scheme, masking strategy, sources of training corpora, length of training sequences. We will release our NEZHA models as well as the source code to the community.

2 Pre-training NEZHA Models

In this section, we present our NEZHA model in details. Section 2.1 presents the preliminaries of the BERT model and the positional encoding schemes. Section 2.2 presents the functional relative positional encoding adopted in our model. Section 2.3, 2.4 and 2.5 introduce the three techniques used in our pre-training, i.e., whole word masking, mixed precision training and the LAMB optimizer.

2.1 Preliminaries: BERT Model & Positional Encoding

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained language model, which is a stack of Transformer encoders. Each Transformer encoder is a multi-head self-attention laryer followed by a position-wise feed-forward network. It uses residual connections around each sub-layer, followed by a layer normalization. We refer the reader to [9] for more details of the Transformer architecture. Each sample in the training data of BERT is a pair of sentences. In each sample, 12% tokens are masked and 1.5% tokens are randomly replaced by another token in the vocabulary. Besides, in the training set, each sample (containing sentences *A* and *B*) is constructed as follows. 50% of the times, *B* is actually the next sentence of *A* and 50% times *B* is a random sentence from the corpus, which is not the next sentence of *A*. In the pre-training phase, BERT has two tasks. One is the masked language modeling (MLM), which aims to predict the masked tokens from other tokens. The second pre-training task is the next sentence prediction (NSP). It predicts if the second sentence in each training sample is the next sentence of the first sentence or not. In some sense, BERT can be regarded as a denoising auto-encoder, since one of its training objectives is to recover the data with noises added.

In Transformer, each attention head operates on a sequence of tokens $x = (x_1, x_2, \ldots, x_n)$, where $x_i \in \mathbb{R}^{d_x}$, and outputs a sequence $z = (z_1, z_2, \ldots, z_n)$ of the same length, where $z_i \in \mathbb{R}^{d_z}$. Each attention head has three parametric matrices W^K, W^Q and $W^V \in \mathbb{R}^{d_x \times d_z}$ to be learned. The output z_i is calculated as follows.

$$z_i = \sum_{j=1}^{n} \alpha_{ij}(x_j W^V). \tag{1}$$

The attention score α_{ij} between the hidden states in position i and position j is computed by using a softmax function:

$$\alpha_{ij} = \frac{\exp e_{ij}}{\sum_k \exp e_{ik}}, \tag{2}$$

where e_{ij} is the scaled dot product between the linear transformations of the input elements:

$$e_{ij} = \frac{(x_i W^Q)(x_j W^K)^T}{\sqrt{d_z}}. \tag{3}$$

Since the multi-head attention in Transformer (and BERT) is permutation invariant, and thus not sensitive to the word order. Therefore, [9] incorporates an absolute positional encoding for each position, which is an embedding vector and added to the token embedding directly. Later on, [11] proposes a parametric relative positional encoding for Transformer. In the relative positional encoding scheme, the computation of the attention scores involves a parametric embedding regarding the relative distance between the two positions. Specifically, it modifies the computation of the output z_i in equation 1 and the e_{ij} in equation 3 as follows:

$$z_i = \sum_{j=1}^{n} \alpha_{ij}(x_j W^V + a_{ij}^V), \tag{4}$$

$$e_{ij} = \frac{(x_i W^Q)(x_j W^K + a_{ij}^K)^T}{\sqrt{d_z}}. \tag{5}$$

In the two equations above, $a_{ij}^V, a_{ij}^K \in \mathbb{R}^{d_z}$ are two vectors with the relative position between i and j encoded, and they are shared across all attention heads. Transformer-XL [12] and XLNet [6] implement the relative positional encoding with a different formulation. We refer the reader to their paper for more details.

2.2 Functional Relative Positional Encoding

In the current version of NEZHA, we employ functional relative positional encoding, where the computation of the outputs and attention scores involves sinusoidal functions of their relative position. This idea is inspired by the functional absolute positional encoding adopted in Transformer [9]. Specifically, in our model, a_{ij}^V and a_{ij}^K are both derived from sinusoidal functions and fixed during the model training. In the remainder of this technical report, we denote a_{ij} to present the formulation of a_{ij}^V and a_{ij}^K for clarity. Consider the dimension $2 \cdot k$ and the dimension $2 \cdot k + 1$ of a_{ij} respectively,

$$a_{ij}[2k] = \sin((j-i)/(10000^{\frac{2 \cdot k}{d_z}})), \tag{6}$$

$$a_{ij}[2k+1] = \cos((j-i)/(10000^{\frac{2 \cdot k}{d_z}})). \tag{7}$$

That is, each dimension of the positional encoding corresponds to a sinusoid, and the sinusoidal functions for different dimensions have different wavelengths. In the above equations, d_z is equal to the hidden size per head of the NEZHA model (i.e., the hidden size divided by the number of heads). The wavelengths form a geometric progression from 2π to $10000 \cdot 2\pi$. We choose the fixed sinusoidal functions mainly because it may allow the model to extrapolate to sequence lengths longer than the ones encountered during training.

2.3 Whole Word Masking

In the vanilla BERT, each token or Chinese character is masked randomly. In [8], whole word masking (WWM) strategy is found to be more effective than random masking for training BERT. In WWM, once a Chinese character is masked, the other characters belonging to the same Chinese word are all masked together. In implementing WWM for NEZHA, we used a tokenization tool Jieba[2] for the Chinese word segmentation (i.e., finding the boundaries of the Chinese words). In the WWM training data, each sample contains several masked Chinese words, and the total number of masked Chinese characters is roughly 12% of its length and 1.5% randomly replaced characters.

2.4 Mixed Precision Training

In the pre-training of our NEZHA models, we adopt the technique of mixed precision training [13]. The technique can speed up the training by 2-3 times and also reduce the space consumption of the model, as a result of which, a larger batch size could be utilized.

Conventionally, the training of deep neural networks uses FP32 (i.e., single-precision float point format) to present all the variables (including the model parameters and gradients) involved in the training. Mixed precision training [13]

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

adopts *mixed-precision* in the training. Specifically, it maintains a single-precision copy (called *Master Weights*) of the weights in the model. In each training iteration, it rounds the Master Weights into FP16 (i.e., half-precision float point format) and performs the forward and backward pass with the weights, activations and gradients stored in FP16 format. Finally, it converts the gradients into FP32 format and updates the Master Weights by using the FP32 gradients.

2.5 LAMB Optimizer

The LAMB optimizer [14] is designed for the large batch-size synchronous distributed training of deep neuron networks. Training DNN with large mini-batches is an effective method to speed up the training. However, without careful tuning of the schedule of the learning rate, the performance could be largely harmed when the batch size is beyond a certain threshold. Instead of hand-tuning of the learning rate, the LAMB optimizer employs a general adaptation strategy and meanwhile provides insight into the convergence by theoretical analysis. The optimizer speeds up the training of BERT by using a very large batch size (up to more than 30k in [14]) without incurring a loss of the performance and even obtains the state-of-the-art performance in many tasks. Remarkably, the training time of BERT is significantly reduced from 3 days to 76 minutes.

3 Experiments

In this section, we report the experimental results on pre-training our NEZHA models for Chinese text and finetuning on Chinese NLU downstream tasks. It should be noted that the training techniques are not limited to Chinese and can be readily applied to other languages.

3.1 Experimental Setting

Datasets We adopt three Chinese corpora for pre-training the NEZHA models:

- Chinese Wikipedia [3]. Chinese Wikipedia is a Chinese-language encyclopedia containing 1,067,552 articles. We downloaded the latest Chinese Wikipedia dump and cleaned the raw data with the tool named WikiExtractor[4]. The cleaned corpus contains both simplified and traditional Chinese and has roughly 202M tokens.
- Baidu Baike [5]. We crawled webpages from the Baidu Baike, which is a Chinese-language, collaborative, web-based encyclopedia owned and produced by the Chinese search engine Baidu. As of August 2018, Baidu Baike has more than 15.4 million articles. The cleaned corpus contains 4,734M tokens.
- Chinese News. We crawled Chinese News corpus from multiple news websites (e.g., Sina News). The cleaned corpus contains 5,600M tokens.

For each corpus above, we prepared two versions of the pre-training data for NEZHA. The first version is processed the same as that in [1], which contains 12% masked Chinese characters and 1.5% randomly replaced Chinese characters. We used tools provided by the BERT Github project [6] to convert the text data into the pre-training examples. The second version is based on the whole word masking (WWM) strategy. We created the WWM pre-training examples with the Chinese word segmenter Jieba for identifying the boundaries of Chinese words. In the WWM examples, each sample contains several masked Chinese words, and the total number of masked Chinese characters is roughly 12% of its length and 1.5% randomly replaced characters. Table 1 summarizes the statistics of the datasets for several pre-trained models.

Pre-training Details We train the NEZHA models on 10 servers on Huawei Cloud [7], each of which has 8 NVIDIA Tesla V100 GPUs with 32GB memory. The distributed training algorithm is the *Ring-AllReduce*[8] and was employed with the framework named Horovod [15]. We trained each model from scratch and terminated the training when the training loss converged. For the $\text{NEZHA}_{\text{BASE}}$ models, we set the maximum learning rate to be $1.8e-4$ (with 1800 warm-up steps and linear decay). The batch size on each GPU is 180 and thus the total batch size is 180 * 8 * 10 = 14400. For the $\text{NEZHA}_{\text{LARGE}}$ models, we set the maximum learning rate to be $1e-4$ (with 1800 warm-up steps and polynomial decay). The batch size on each GPU is 64, and thus the total batch size is 64 * 8 * 10 = 5120. In addition, we adopted the mixed-precision training using FP16 [13] in the pre-training phase.

[3] https://zh.wikipedia.org/wiki/
[4] https://github.com/attardi/wikiextractor
[5] https://baike.baidu.com/
[6] https://github.com/google-research/bert
[7] https://www.huaweicloud.com/product/modelarts.html
[8] https://github.com/baidu-research/baidu-allreduce

Table 1: Configurations of Chinese pre-trained language models

Model	Pre-Training Corpora	#Tokens	Vocabulary size	Activation function	Hidden Size/#Layers	#Heads
$\text{BERT}_{\text{BASE}}$ $\text{BERT}_{\text{BASE}}$-WWM	Wikipedia	202M	21,128	GeLU	768/12	12
$\text{ERNIE-Baidu}_{\text{BASE}}$ 1.0	Wikipedia+Baike+Tieba	9,388 M	18,000	ReLU	768/12	12
$\text{ERNIE-Baidu}_{\text{BASE}}$ 2.0	Baike+News+Dialog	14,988M			768/12	12
$\text{ERNIE-Baidu}_{\text{LARGE}}$ 2.0					1024/24	16
$\text{NEZHA}_{\text{BASE}}$	Wikipedia+Baike+News	10,536M	21,128	GeLU	768/12	12
$\text{NEZHA}_{\text{LARGE}}$					1024/24	16

Table 2: Pre-training Techniques Adopted in Chinese pre-trained language models (MLM: Masked Language Modeling, NSP: Next Sentence Prediction, WWM: Whole Word Masking, KM: Knowledge Masking, SR: Sentence Reordering, SD: Sentence Distance, DR: Discourse Relation, IR: IR Relevance, PAPE: Parametric Absolute Position Encoding, FRPE: Functional Relative Position Encoding)

Model	Pre-Training Tasks			Training Precision	Optimizer	Position Encoding
	Word-Aware	Sentence-Aware	Semantic-Aware			
$\text{BERT}_{\text{BASE}}$	MLM	NSP	-	Single Precision (FP32)	ADAM	PAPE
$\text{BERT}_{\text{BASE}}$-WWM	MLM (WWM)	NSP			LAMB	
$\text{ERNIE-Baidu}_{\text{BASE}}$ 1.0	MLM (KM)	NSP	-	Single Precision (FP32)	ADAM	PAPE
$\text{ERNIE-Baidu}_{\text{BASE}}$ 2.0	MLM (KM)	SR & SD	DR & IR	Mixed Precision	ADAM	PAPE
$\text{ERNIE-Baidu}_{\text{LARGE}}$ 2.0						
$\text{NEZHA}_{\text{BASE}}$	MLM (WWM)	NSP	-	Mixed Precision	LAMB	FRPE
$\text{NEZHA}_{\text{LARGE}}$						

3.2 Experimental Results

In the experiment, we compared NEZHA models with the state-of-the-art Chinese pre-trained language models: Goolge's BERT [1] for Chinese, BERT-WWM [8] and ERNIE-Baidu [3, 4]. Their model configurations are shown in Table 1. We also summarize pre-training techniques adopted in each Chinese pre-trained language models in Table 2. Note that ERNIE-Baidu has three different versions, which are $\text{ERNIE-Baidu}_{\text{BASE}}$ 1.0, $\text{ERNIE-Baidu}_{\text{BASE}}$ 2.0 and $\text{ERNIE-Baidu}_{\text{LARGE}}$ 2.0. $\text{ERNIE-Baidu}_{\text{BASE}}$ and $\text{ERNIE-Baidu}_{\text{LARGE}}$2.0 introduced many different pre-training tasks and we refer the readers to thier papers for the details of these tasks. As shown in the table, the unique technique in our models is the functional relative position encoding. We test the performances of the pre-trained models by finetuning on a variety of natural language understanding (NLU) tasks, which are listed as follows. The hyperparameters of finetuning each task are shown in Table 3.

- **CMRC** (Chinese Machine Reading Comprehension 2018) [16]: A machine reading comprehension task that returns an answer span in a given passage for a given question.
- **XNLI** (Cross-lingual Natural Language Inference) [17]: The Chinese portion of XNLI, which is a version of MultiNLI where the dev and test sets have been translated (by humans) into 15 languages. XNLI is a natural language inference task. The goal of this task is to predict if the second sentence is a contradiction, entailment or neutral to the first sentence.
- **LCQMC** (Large-scale Chinese Question Matching Corpus) [18]: A sentence pair matching task. Given a pair of sentences, the task is to determine if the two sentences are semantically equivalent or not.
- **PD-NER** (People's Daily Named Entity Recognition) [9]: A sequence labeling task that identifies the named entities from text. The corpus is from *People's Daily*, a Chinese News Media.
- **ChnSenti** (Chinese Sentiment Classification) [10]: A binary classification task which predicts if the sentiment of a given sentence is positive or negative.

[9] https://github.com/ProHiryu/bert-chinese-ner

[10] https://github.com/pengming617/bert_classification

Table 3: Hyperparameters used in finetuning downstream tasks. (SL: sequence length; LR stands: learning rate.)

Task Name	Batch Size (BASE/LARGE)	SL	LR	Epochs	#Train	#Dev	#Test	Domain
CMRC	16/72	384	3e-5	2	10K	3.2K	-	Wikipedia
XNLI	64/32	128	3e-5	3	392K	2.5K	2.5K	General
LCQMC	64/32	128	3e-5	5	240K	8.8K	12.5K	QA
PD-NER	64/16	256	3e-5	5	51K	4.6K	68	News
ChnSenti	64/16	256	3e-5	10	9.6K	1.2K	1.2K	General

Table 4: Results of pre-trained models on downstream Chinese NLU tasks.

Model	CMRC		XNLI		LCQMC		PD-NER		ChnSenti	
	EM	F1	Dev	Test	Dev	Test	Dev	Test	Dev	Test
BASE MODELS										
$\text{BERT}_{\text{BASE}}$	64.06	85.01	78.75	77.27	89.04	87.61	96.53	98.58	94.91	95.42
$\text{BERT}_{\text{BASE}}$-WWM	64.96	85.79	78.79	78.44	89.19	87.16	96.86	98.58	94.67	94.58
$\text{BERT}_{\text{BASE}}$-WWM (in [8])	66.30	85.60	79.00	78.20	89.40	87.00	95.30	65.10	95.10	95.40
ERNIE-$\text{Baidu}_{\text{BASE}}$ 1.0 (in [3])	65.10	85.10	79.9	78.4	89.70	87.40	-	-	95.20	95.40
ERNIE-$\text{Baidu}_{\text{BASE}}$ 2.0 (in [4])	**69.10**	**88.60**	81.20	**79.70**	**90.90**	**87.90**	-	-	**95.70**	95.50
$\text{NEZHA}_{\text{BASE}}$ (ours)	67.07	86.35	**81.37**	79.32	89.98	87.41	97.22	**98.58**	94.74	95.17
$\text{NEZHA}_{\text{BASE}}$-WWM (ours)	67.82	86.25	81.25	79.11	89.85	87.10	**97.41**	98.35	94.75	**95.84**
LARGE MODELS										
ERNIE-$\text{Baidu}_{\text{LARGE}}$ 2.0 (in [4])	**71.50**	**89.90**	**82.60**	81.00	**90.90**	87.90	-	-	**96.10**	95.80
$\text{NEZHA}_{\text{LARGE}}$ (ours)	68.10	87.20	81.53	80.44	90.18	87.20	**97.51**	**97.87**	95.92	95.83
$\text{NEZHA}_{\text{LARGE}}$-WWM (ours)	67.32	86.62	82.21	**81.17**	90.87	**87.94**	97.26	97.63	95.75	**96.00**

We show the comparison results of different pre-trained models on the aforementioned tasks in Table 4. Among the groups of both BASE models and LARGE models, either ERNIE-Baidu 2.0 or NEZHA achieves the best performances. Note that the part of the results are directly copied from the original papers [8, 4]. Due to the possible differences in the experimental setting or finetuning methods, the comparison may not be entirely fair. We notice that there is consistent gaps between our implementation and the results reported in [8, 4] on the CMRC task. Once the ERNIE-Baidu 2.0 Chinese models are released, we will evaluate them under the same setting and update this report.

3.3 Ablation Study

In this section, we study the effectiveness of the data and different techniques for training NEZHA, which are listed as follows.

- **Positional Encoding:** the effectiveness of the functional relative positional encoding (FRPE) employed in our work compared with the parametric absolute positional encoding (PAPE) and parametric relative positional encoding (PRPE) adopted in the existing studies.
- **Masking Strategy:** the effect of the whole word masking (WWM) on the performance of the pre-trained models.
- **Training Sequence Length:** the impact of the training with longer sequences.
- **Training Corpora:** the impact of the source of the training data.

With the above objectives, we compare the performances of several variants of $\text{NEZHA}_{\text{BASE}}$ model, as shown in Table 5. The results demonstrate that the techniques mentioned above generally have positive contributions to the downstream tasks, where functional relative positional encoding shows a notable advantage compared with other positional encoding methods. For instance, we can see that when trained with a maximum of 128 tokens, the model using the absolute positional encodings performs significantly worse than those using relative positional encodings on the CMRC task, where the input passages can be much longer.

Table 5: Ablation studies. (PAPE: parametric absolute positional encoding; PRPE: parametric relative positional encoding; FRPE: functional relative positional encoding; WWM: whole word masking; SL: sequence length.)

Model	CMRC		XNLI		LCQMC		PD-NER		ChnSenti	
	EM	F1	Dev	Test	Dev	Test	Dev	Test	Dev	Test
$\text{NEZHA}_{\text{BASE}}$										
News, PAPE, SL:128	37.96	58.40	78.79	77.72	89.31	86.74	94.87	98.10	94.17	95.67
News, PRPE, SL:128	65.26	86.17	79.18	77.98	89.21	86.92	96.93	98.12	94.67	95.08
News, FRPE, SL:128	65.95	86.46	79.96	78.32	89.40	87.23	96.69	98.10	95.58	95.75
News, FRPE, SL:512	67.79	86.60	80.57	79.52	90.06	86.73	97.04	97.62	95.09	95.08
News+Wiki+Baike, FRPE, SL:128	66.95	86.41	81.25	79.06	89.83	87.13	97.21	97.41	95.25	94.42
News+Wiki+Baike, FRPE, WWM, SL:128	67.82	86.25	81.25	79.11	89.85	87.10	97.41	98.35	94.75	95.84
News+Wiki+Baike, FRPE, WWM, SL:512	66.45	86.16	80.96	79.86	89.64	86.18	96.79	98.10	95.08	95.42

4 Conclusion

In the technical report, we have presented our practice on training the large scale pre-trained language models NEZHA on Chinese corpora. We have employed an effective functional relative positional encoding scheme, which leads to notable improvement over the other positional encodings. The pre-training of the NEZHA models also integrates several techniques, including whole word masking strategy, mixed precision training, and the LAMB optimizer. Experiments show that our models can achieve state-of-the-art performances on several Chinese natural language understanding tasks. In the future, we plan to continue the work on improving NEZHA on Chinese and other languages and extend the applications of NEZHA to more scenarios.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[2] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237, 2018.

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

[4] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. Ernie 2.0: A continual pre-training framework for language understanding. *arXiv preprint arXiv:1907.12412*, 2019.

[5] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. Ernie: Enhanced language representation with informative entities. *arXiv preprint arXiv:1905.07129*, 2019.

[6] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*, 2019.

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

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

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[10] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[11] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, 2018.

[12] Zihang Dai, Zhilin Yang, Yiming Yang, William W Cohen, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

[13] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. *arXiv preprint arXiv:1710.03740*, 2017.

[14] Yang You, Jing Li, Jonathan Hseu, Xiaodan Song, James Demmel, and Cho-Jui Hsieh. Reducing bert pre-training time from 3 days to 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.

[15] Alexander Sergeev and Mike Del Balso. Horovod: fast and easy distributed deep learning in TensorFlow. *arXiv preprint arXiv:1802.05799*, 2018.

[16] Yiming Cui, Ting Liu, Li Xiao, Zhipeng Chen, Wentao Ma, Wanxiang Che, Shijin Wang, and Guoping Hu. A span-extraction dataset for chinese machine reading comprehension. *arXiv preprint arXiv:1810.07366*, 2018.

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

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