KNOWLEDGE DISTILLATION FROM BERT IN PRE-TRAINING AND FINE-TUNING FOR POLYPHONE DISAMBIGUATION

Hao Sun[1], Xu Tan[2], Jun-Wei Gan[3], Sheng Zhao[3], Dongxu Han[3], Hongzhi Liu[1], Tao Qin[2] and Tie-Yan Liu[2]

[1]Peking University, Beijing
[2]Microsoft Research, Beijing
[3]Microsoft STC Asia, Beijing
sigmeta@pku.edu.cn, xuta@microsoft.com, junwg@microsoft.com, Sheng.Zhao@microsoft.com, dohan@microsoft.com, liuhz@pku.edu.cn, taoqin@microsoft.com, tyliu@microsoft.com

ABSTRACT

Polyphone disambiguation aims to select the correct pronunciation for a polyphonic word from several candidates, which is important for text-to-speech synthesis. Since the pronunciation of a polyphonic word is usually decided by its context, polyphone disambiguation can be regarded as a language understanding task. Inspired by the success of BERT for language understanding, we propose to leverage pre-trained BERT models for polyphone disambiguation. However, BERT models are usually too heavy to be served online, in terms of both memory cost and inference speed. In this work, we focus on efficient model for polyphone disambiguation and propose a two-stage knowledge distillation method that transfers the knowledge from a heavy BERT model in both pre-training and fine-tuning stages to a light-weight BERT model, in order to reduce online serving cost. Experiments on Chinese and English polyphone disambiguation datasets demonstrate that our method reduces model parameters by a factor of 5 and improves inference speed by 7 times, while nearly matches the classification accuracy (95.4% on Chinese and 98.1% on English) to the original BERT model.

Index Terms— Polyphone Disambiguation, Knowledge Distillation, Pre-training, Fine-tuning, BERT

1. INTRODUCTION

Grapheme-to-Phoneme (G2P) conversion is an important module in automatic speech recognition and text-to-speech systems, aiming to predict the correct pronunciation of a word given its orthography. For character-based languages such as Chinese and Japanese, the pronunciation of a character is fixed or with a few fixed candidates when the character is a polyphone. Therefore, the G2P conversion task in these languages is to select the correct pronunciation of a polyphone from the candidates [1, 2]. For alphabetic languages such as English and French, the main challenge of G2P conversion is how to predict the pronunciations of out-of-vocabulary (OOV) words [3, 4, 5]. Nevertheless, there still exist polyphonic words in these languages.

Polyphone disambiguation, which aims to pick up the right pronunciation of a polyphonic word, is of great importance in text-to-speech system. Considering the number of candidate pronunciations of a polyphonic word is finite, polyphone disambiguation can be regarded as a classification task. Previous works on polyphone disambiguation include rule based method [6] and learning based method such as maximum entropy [2, 7], decision tree [8, 9] and LSTM [10]. Although these approaches achieve good performance, both of them have some limitations: 1) Rule based methods require human expertise which is hard to obtain and costly. At the same time, it cannot cover all the patterns, and thus cannot generalize well to unseen patterns. 2) Learning based methods can extract statistical knowledge from the data and outperform the rule based approach. However, traditional machine learning methods require complex feature design and lack fitting capacity compared with neural network. While LSTM models can achieve better results, neural models are usually data-hungry, which prevents them from achieving higher accuracy.

Recently, pre-training and fine-tuning, such as BERT [11], GPT [12] and MASS [13], have achieved great success in many language understanding and generation tasks by transferring knowledge from pre-trained deep transformer model [14] to downstream tasks. Based on the carefully designed unsupervised pre-training loss, e.g., masked language model, the pre-trained model can extract semantic information, understand meanings or even capture common-sense knowledge from large scale corpus. Polyphone disambiguation is a typical language understanding task that requires the understanding of the contextual semantic meaning for polyphone classification, and also suffers from limited training data, which will benefit from BERT pre-training. Therefore, we can pre-train the BERT model on a large scale unlabeled corpus and fine-tune it on the polyphone disambiguation

dataset to boost the accuracy.

However, BERT model is usually of huge model size, i.e., more than hundreds of millions of parameters, which brings challenges in memory cost and inference latency for the online serving in text-to-speech systems. In this work, we propose to distill the knowledge from a standard BERT model into a smaller BERT model, which can greatly reduce the number of model parameters, and thus memory cost and inference latency. Unlike the previous knowledge distillation in a single stage [15], we redesign the knowledge distillation in the pre-training and fine-tuning pipeline, with a two-stage distillation method that transfers the knowledge from BERT in both pre-training and fine-tuning stages to ensure the accuracy of smaller model. Experiments on the Chinese and English polyphone disambiguation datasets demonstrate that our method reduces the model parameter of BERT by a factor of 5 and improves the inference speed by 7 times, while maintains the accuracy with the original BERT model.

Our contributions are listed as follows: (1) We are the first to introduce the pre-trained deep bidirectional Transformer model (BERT) into polyphone disambiguation task. (2) We propose a two-stage knowledge distillation method in pre-training and fine-tuning stages to reduce the number of parameters as well as ensure the accuracy of smaller BERT model. (3) Our method achieves state-of-the-art accuracy on the Chinese and English polyphone disambiguation datasets, with the classification accuracy of 95.4% (nearly match the accuracy of BERT base model, and 6.9% better than the conventional LSTM model), while reduces the number of model parameters of BERT by a factor of 5 and improve the inference speed by 7 times.

2. BACKGROUND

In this section, we introduce the background of this work, including polyphone disambiguation task, pre-training method, and knowledge distillation.

2.1. Polyphone Disambiguation

Polyphone phenomenon, which means a character or a word has more than one pronunciation, is quite common in some languages such as Chinese. For example, the Chinese character "没" has two pronunciations, as shown in Table 1. In the alphabetic languages such as English, polyphone phenomenon also exists. For example, the word "conflict" is read as "ˈkɑːnflɪkt" when it is a noun, and is read as "kənˈflɪkt" when it is a verb.

Polyphone disambiguation aims to pick up the right pronunciation of a polyphonic word [1, 16], and is critical to a high quality text-to-speech synthesis system. As the number of candidate pronunciations of a word is usually fixed, polyphone disambiguation is regarded as a classification task [10]. Considering that the pronunciation of a word is usually decided by its part-of-speech (POS) and context, the early approaches of polyphone disambiguation are rule based [1, 6, 8]. Afterwards, machine learning methods [2, 7, 8, 9, 17] can extract rules automatically based on labeled training data. In recent years, neural networks such as LSTM have improved the accuracy of polyphone disambiguation significantly [10, 18].

Table 1. An example of Chinese polyphonic character "没" with two different pronunciations and the corresponding English meanings.

Word	Pronunciation (Pin Yin)	Meaning in English
没有	mei2	don not have
淹没	mo4	submerge

However, these methods either require handcrafted feature design which is costly, or relies on large amount of training data which is often scarce. Inspired by the recent large scale unsupervised pre-training method for language understanding, in this work, we leverage the pre-training and fine-tuning pipeline [11, 12, 13] for ploy phone disambiguation to improve the accuracy.

2.2. Pre-traning and Fine-tuning

Recently, pre-training and fine-tuning has demonstrated unprecedented effectiveness for learning universal language representations on unlabeled corpus and improving the downstream tasks in natural language processing. Some of the most prominent models include BERT [11], GPT [12] and MASS [13]. Among these models, BERT is designed for language understanding tasks [11, 19, 20] and is very suitable for polyphone disambiguation which leverages token-level classification.

BERT introduces two pre-training methods: masked language model and next sentence prediction [11], which gives the model good representations at both token level and sentence level. After that, BERT takes a fine-tuning strategy for downstream tasks by jointly fine-tuning the pre-trained parameters and the minimal task-specific parameters. BERT is trained on a deep bidirectional Transformer [14], with hundreds of millions of model parameters, which requires large memory cost and inference latency, and hinders the usage in online service. In this paper, we leverage knowledge distillation to compress the BERT model into a much smaller model, in order to satisfy the online serving requirements without loss of accuracy.

2.3. Knowledge Distillation

Knowledge distillation was first introduced by [21] for model compression, where a light student model can approximate the accuracy of a heavy and cumbersome teacher model by distilling the knowledge from the teacher model. [22] first applied knowledge distillation on neural networks, and then a

lot of works expand the usage of knowledge distillation to a variety of tasks, such as image classification [23, 24, 25] and natural language processing [26, 27, 28]. In knowledge distillation, the student model usually takes the soft outputs of the teacher network as training targets. The conventional one-hot labels aim to project the samples in each class into a single point in the label space, while the soft labels project the samples into a continuous distribution. Compared with one-hot labels, soft outputs provide extra supervisions of intra-class and inter-class similarities learned by the teacher model [29]. By learning from the soft targets which contain intra-class variation and inter-class relationships, the knowledge can be better transferred from heavy models to light-weight student models.

Here we give the formulation of knowledge distillation on a multi-class classification task. Given the training data (x, y) with $|\mathcal{V}|$ classes, where x is the input and y is the label, the original negative log-likelihood loss is

$$\mathcal{L}_{NLL}(\theta) = -\sum_{k=1}^{|\mathcal{V}|} \mathbb{1}\{y = k\} \log P(y = k|x; \theta), \quad (1)$$

where $\mathbb{1}$ is the indicator function indicating if y equals to k, P is the output distribution of the model θ. For knowledge distillation, given a teacher model θ_T, the student model learns to match the outputs of the teacher model [26] with cross entropy loss

$$\mathcal{L}_{KD}(\theta; \theta_T) = -\sum_{k=1}^{|\mathcal{V}|} Q\{y = k|x; \theta_T\} \log P(y = k|x; \theta), \quad (2)$$

where Q is the probability distribution of the teacher model θ_T and is fixed during training.

In this work, unlike the conventional methods, we introduce knowledge distillation into the pre-training and fine-tuning pipeline, not only distilling in the fine-tuning stage but also in the pre-training stage to ensure the accuracy of the student model.

3. KNOWLEDGE DISTILLATION IN PRE-TRAINING AND FINE-TUNING

Similar to the pre-training and fine-tuning pipeline in BERT, we also conduct similar strategy for the polyphone disambiguation task. In order to ensure the accuracy with smaller BERT model which is online friendly, we apply knowledge distillation in both pre-training and fine-tuning stages.

3.1. Knowledge Distillation in Pre-training

BERT [11] introduces two pre-training losses: masked language model and next sentence prediction, to help the model learn good representations at both token level and sentence level for various downstream tasks. Considering that polyphone disambiguation is a token-level classification task, which mainly leverages the contextual information in the current sentence, we simplify the pre-training procedure with only masked language model loss. We follow similar masking procedure as used in BERT: 15% tokens are randomly masked, where 80% are masked with the token "[MASK]", 10% are masked with a random token, and the rest 10% are remained unchanged.

During pre-training, instead of learning from ground truth labels, we distill the knowledge from the pre-trained BERT model. For each sentence, BERT teacher model generates the probability distributions for the masked tokens, which are taken as soft labels to teach the light-weight BERT model. For each sentence, both the teacher and student BERT model share the same masked strategy and masked tokens. The loss function for knowledge distillation in pre-training stage is

$$\mathcal{L}_{KD\_Pre}(\theta) = -\sum_{x\in\mathcal{D}} \sum_{m\in M^x} \sum_{k=1}^{|\mathcal{V}|} Q(x_m = k|x; \theta_T) \times \log P(x_m = k|x; \theta), \quad (3)$$

where $\mathcal{D}$ is the sentence corpus for pre-training, M^x is the set of indices of the masked tokens in sentence x and x_m is the corresponding prediction of the masked token, $\mathcal{V}$ is the vocabulary of all tokens, Q is the probability distribution of the BERT teacher model θ_T which is fixed during training, and P is the output distribution of the small BERT student model θ.

3.2. Knowledge Distillation in Fine-tuning

After pre-training, the BERT student model is fine-tuned on the training data of polyphone disambiguation. To ensure the accuracy, we also conduct knowledge distillation to transfer the knowledge from the BERT teacher model to the student model. Therefore, we first fine-tune the BERT teacher model θ_T on the polyphone disambiguation task with original negative log-likelihood loss to obtain the teacher model θ'_T. The loss function for the knowledge distillation in the fine-tuning stage is

$$\mathcal{L}_{KD\_Fine}(\theta') = -\sum_{(x,y)\in\mathcal{D}'} \sum_{w\in W^x} \sum_{k\in\mathcal{C}} Q(y_w = k|x; \theta'_T) \times \log P(y_w = k|x; \theta'), \quad (4)$$

where $\mathcal{D}'$ is the supervised training data for polyphone disambiguation, W^x is the set of indices of the polyphonic words in the training sentence x and y_w is the classification output of the corresponding polyphonic word, $\mathcal{C}$ is the set of classes for different pronunciations, Q is the probability distribution of the fine-tuned BERT model θ'_T and P is the output distribution of the small BERT model θ', where θ' has been

pre-trained and only differs from θ in Equation 3 in the task-specific output layer.

3.3. Fine-tuning Design for Polyphone Disambiguation

Polyphone disambiguation is usually regarded as a classification task, where each polyphonic word has a fixed number of classes, each class corresponding to a pronunciation. In order to support the classification of multiple polyphonic words in one model, previous methods [10] suffer from a problem that the same pronunciation of different characters is treated as one class [10]. For example, "觉" and "角" share the same pronunciation "jue2" and thus share the same class in prediction. Since the same pronunciation of different words has no semantic relation but are tied together, which will influence the prediction accuracy.

To solve the problem mentioned above, we make careful design during fine-tuning to split the class of the same pronunciation for different characters into different classes. For example, "觉" and "角" share the same pronunciation "jue2", which will be treated as two different classes "觉jue2" and "角jue2". We further introduce class mask in the softmax classification to reduce the influence of different characters on each other.

The target labels in polyphone disambiguation are all the pronunciations of all polyphonic words, as used in [10]. The negative log-likelihood loss on the ground truth label is

$$\mathcal{L}_{NLL}(\theta') = -\sum_{(x,y)\in\mathcal{D}'}\sum_{w\in W^x}\sum_{k\in\mathcal{C}}\mathbb{1}\{y_w=k\} \times \log P(y_w=k|x;\theta'), \tag{5}$$

where θ', $\mathcal{D}'$ is the model parameter and training data, W^x is the set of indices of the polyphonic words in the training sentence x and y_w is the classification output of the corresponding polyphonic word, $\mathcal{C}$ is the vocabulary for all the pronunciations, $\mathbb{1}$ is the indicator function.

However, in the polyphone disambiguation task, the number of the candidate pronunciations of a character are far less than the whole vocabulary. The loss in Equation 5 will adjust the probabilities of other irrelevant classes, where different characters may affect each other. We address this problem by masking the irrelevant classes and only calculating the loss of the candidate classes. The loss with class mask is:

$$\mathcal{L}_{NLL\_CM}(\theta') = -\sum_{(x,y)\in\mathcal{D}'}\sum_{w\in W^x}\sum_{k\in\mathcal{C}_w}\mathbb{1}\{y_w=k\} \times \log P(y_w=k|x;\theta'), \tag{6}$$

where $\mathcal{C}_w$ is the set of classes for the candidate pronunciations of the word w. With knowledge distillation, the loss becomes

$$\mathcal{L}_{KD\_CM}(\theta') = -\sum_{(x,y)\in\mathcal{D}''}\sum_{w\in W^x}\sum_{k\in\mathcal{C}_w}Q(y_w=k|x;\theta'_T) \times \log P(y_w=k|x;\theta'), \tag{7}$$

where Q is the probability distribution of the teacher model θ'_T, and $\mathcal{D}''$ is the training data including $\mathcal{D}'$ and additional unlabeled data.

3.4. Two-stage Knowledge Distillation

We summarize the whole framework of the two-stage knowledge distillation for polyphone disambiguation, as shown in Algorithm 1.

Algorithm 1 Two-stage Knowledge Distillation

Input: Pre-training corpus $\mathcal{D}$, polyphone disambiguation traing data $\mathcal{D}'$, and $\mathcal{D}''$ which is $\mathcal{D}'$ with additional unlabeled data.

1: Pre-train BERT base model θ_T (teacher) on corpus D with only masked langauge model loss.
2: Fine-tune BERT base model θ'_T (teacher) on polyphone disambiguation training data $\mathcal{D}'$ following Equation 5 .
3: Pre-train BERT small model θ (student) on corpus D with knowledge distillation following Equation 3.
4: Fine-tune BERT small model θ' (student) on polyphone disambiguation training data $\mathcal{D}''$ with knowledge distillation following Equation 7.

4. EXPERIMENTAL SETUP

In this section, we introduce the experimental setup in order to verify the effectiveness of the proposed method. We first introduce the datasets used, and then describe the model configurations and implementation details.

4.1. Datasets

For pre-training, we crawl our corpus from news and wikipedia. As we evaluate polyphone disambiguation on both Chinese mandarin and English, we crawled 140 million sentences for Chinese and 240 million sentences for English.

For polyphone disambiguation on Chinese mandarin, we choose the 79 most frequent polyphonic characters from our internal datasets. The training set contains 166,185 sentences and the test set contains 43,426 sentences, where one sentence has only one labeled polyphonic character. For English, we choose 14 most frequent polyphonic words from our internal datasets. The training set and test set contain 18,363 and 6,802 sentences respectively.

For knowledge distillation during fine-tuning, we also leverage additional unlabeled data for polyphone disambiguation, which is selected from the pre-training corpus and contains 10,000 sentences for each polyphonic character or word. That is to say, we have 790,000 unlabeled sentences for Chinese and 140,000 unlabeled sentences for English.

4.2. Model Configuration

We choose BERT base model released by [11] as the teacher model, which consists of 12 Transformer layers, with 768 hidden dimension, 12 attention heads, and about 110 millions of parameters in total. For the small BERT model, we follow the similar Transformer architecture with BERT, but with different number of layers and hidden dimensions, as shown in Table 4.

Table 2. Different settings and numbers of parameters of the BERT models.

#Layers	#Hiddens	#Parameters	Reduction
12	768	110M	-
6	512	30M	3x
3	512	20M	5x
3	256	8M	13x

The number of parameters of the model settings in Table 2 have been reduced by about 3 times, 5 times and 13 times respectively, compared with the BERT base model with 110M parameters. Considering that the input token embeddings take up about 10 million for the first two settings and 5 million for the third setting, and embedding lookup will not cost much time, the inference speedup will be larger than the parameter reduction ratio.

4.3. Training and Evaluation

We implement our models on the pytorch-pretrained-BERT [1] codebase using PyTorch. In the pre-training stage of knowledge distillation, the BERT teacher model generates soft labels of the masked tokens for small student models. The batch size is set to 400 sentences with maximum sequence length of 128, the dropout is 0.1 and the learning rate is 1e-4. In the fine-tuning stage, the small student models learn from the fine-tuned BERT teacher model also with knowledge distillation. Meanwhile, unlabeled data is used for further improving the performance of distillation during fine-tuning. The batch size, maximum sequence length and dropout are kept the same as those in pre-training, while the learning rate decreases to 2e-5. During knowledge distillation, the teacher and student models are loaded into memory, and the teacher model generates outputs of every batch online for the student model to learn. Optionally, we can use a weight α to balance the loss between the ground truth labels and the soft labels: $\mathcal{L} = (1-\alpha)\mathcal{L}_{NLL} + \alpha\mathcal{L}_{KD}$. We find that the model performs best when $\alpha = 1$, so we only report the results of knowledge distillation with $\alpha = 1$ in Section 5. Each model is trained on 4 P100 GPUs with the batch size of 100 sentences on each GPU. The classification accuracy on the test set is used for evaluation. Considering the experiment settings used by the release BERT model in [11] are different from ours, such as pre-training corpus, batch size and number of updates, we pre-train a BERT model from scratch with the same number of parameters as the released BERT model, but using our own experiment settings, and fine-tune this pre-training model on polyphone disambiguation and take it as our baseline.

[1] https://github.com/huggingface/pytorch-pretrained-BERT

5. RESULTS

In this section, we first report the results of our method on Chinese polyohone disambiguation dataset, and conduct ablation study to demonstrate the effectiveness of each module in our method, and then also report the results on English dataset.

5.1. Accuracy of Our Method

We report the accuracy of our two-stage knowledge distillation with the best setting of 3 layers and 512 hidden dims in Table 3. We compare our method with several baselines: 1) LSTM [10], which is directly trained on polyphone disambiguation dataset; 2) Transformer [14] with the same configuration with the small BERT model (3 layers and 512 hidden dims) which is also directly trained on the polyphone disambiguation dataset without pre-training; 3) BERT [11], we fine-tuned the released BERT base model on polyphone disambiguation dataset; 4) BERT (reproduce), which is of the same configuration with BERT base but reproduced by ourselves. The results show that 1) BERT has largely improved the accuracy over conventional LSTM and Transformer model (Although Transformer has the same model configuration with the small BERT model, there is still a large accuracy gap due to the effect of pre-training); 2) BERT reproduced by us achieves similar accuracy with BERT released by [11]; 3) Our method reduces the BERT model by about a factor 5 while nearly matches the accuracy to BERT base model, which demonstrates the effectiveness of our method.

Table 3. The accuracy comparison on ployphone disambiguation between our method and other models.

Model	Accuracy	#Parameters
LSTM [10]	88.5%	18M
Transformer	89.3%	20M
BERT [11]	95.9%	110M
BERT (reproduce)	95.5%	110M
Our method	95.4%	20M

5.2. Inference Speedup of Our Method

To analyze the inference speedup of our method, we calculate the inference time cost on the test set on a single P100 GPU for different configurations of BERT model. As shown in Table 4, our method with 3 layers and 512 hidden dims can achieve about 7 times inference speedup compared with BERT base model. The model with 6 layers and 512 hidden dims achieves the same accuracy as our setting, but just has 3 times speedup. The model with 3 layers and 256 hidden dims can achieve about 15 times speedup, but with much lower accuracy. Therefore, we choose 3 layers and 512 hidden dims as our default configuration and call it as small BERT model.

Table 4. Inference speedup for different configurations of BERT models.

#Layers	#Hiddens	Accuracy	Time	Speedup
12	768	95.5 %	137s	-
6	512	95.4 %	38s	3x
3	256	93.9 %	9s	15x
3	512	95.4 %	19s	7x

5.3. Ablation study

In this subsection, we conduct ablation studies on our method, including the effectiveness of the distillation in both pre-training and fine-tuning, as well as the design of class mask in fine-tuning, where we choose the default model setting with 3 layers, 512 hidden dims (small BERT) in the analysis.

5.3.1. Knowledge Distillation in Pre-training

To demonstrate the effectiveness of knowledge distillation in pre-training, we compare the model with and without knowledge distillation during pre-training. We do not use knowledge distillation during fine-tuning for both settings to make fair comparison. As shown in Table 5, knowledge distillation in pre-training improves the accuracy by 0.6%.

Table 5. The accuracy comparison of our small BERT model with and without distillation during pre-training.

Setting	Accuracy
Our method without knowledge distillation	93.6%
Our method with knowledge distillation	94.2%

5.3.2. Knowledge Distillation in Fine-tuning

To demonstrate the effectiveness of knowledge distillation in fine-tuning, we compare the model with and without knowledge distillation during fine-tuning. Before fine-tuning, we use knowledge distillation for both settings during pre-training to make fair comparison. For distillation in fine-tuning, we leverage additional unlabeled data. As shown in Table 6, distillation in fine-tuning further boosts the accuracy by 1.2%.

Table 6. The accuracy comparison of our small BERT model with and without distillation during fine-tuning.

Setting	Accuracy
Our method without knowledge distillation	94.2%
Our method with knowledge distillation	95.4%

5.3.3. Class Mask

We conduct ablation study on class mask (described in Section 3.3) in fine-tuning on polyphone disambiguation. As shown in Table 7, class mask improves the accuracy of polyphone disambiguation by 0.4%, which demonstrates the effectiveness of class mask on reducing the influence between different characters.

Table 7. The accuracy comparison with and without class mask during fine-tuning.

Setting	Accuracy
Our method without class mask	95.0%
Our method with class mask	95.4%

5.4. Results on English dataset

We apply our method on English dataset with 3 layers and 512 hidden dims. As shown in Table 8, out method can nearly match to the accuracy of BERT base model while achieves the same inference speedup (7x) as the Chinese dataset.

Table 8. The accuracy of BERT base model and our small BERT model on English dataset.

Model	BERT (reproduce)	Our method
Accuracy	98.3%	98.1%

6. CONCLUSION

In this work, we have proposed a two-stage knowledge distillation method that transfers the knowledge from a heavy BERT model in both pre-training and fine-tuning stages for polyphone disambiguation, in order to reduce the number of model parameters for online deployment. Experiment results on Chinese and English datasets show that our method reduces BERT base model by a factor of 5 in terms of model parameters and speed up the inference by 7 times while nearly match the accuracy to BERT base model, which demonstrate the effectiveness of our method. For future work, we will design advanced method to further reduce the model size and apply our method on other downstream tasks.

7. REFERENCES

[1] Zi-Rong Zhang, Min Chu, and Eric Chang, "An efficient way to learn rules for grapheme-to-phoneme conversion in chinese," in *International Symposium on Chinese Spoken Language Processing*, 2002.

[2] Xinnian Mao, Yuan Dong, Jinyu Han, Dezhi Huang, and Haila Wang, "Inequality maximum entropy classifier with character features for polyphone disambiguation in mandarin tts systems," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*. IEEE, 2007, vol. 4, pp. IV–705.

[3] Maximilian Bisani and Hermann Ney, "Joint-sequence models for grapheme-to-phoneme conversion," *Speech communication*, vol. 50, no. 5, pp. 434–451, 2008.

[4] Kaisheng Yao and Geoffrey Zweig, "Sequence-to-sequence neural net models for grapheme-to-phoneme conversion," *arXiv preprint arXiv:1506.00196*, 2015.

[5] Hao Sun, Xu Tan, Jun-Wei Gan, Hongzhi Liu, Sheng Zhao, Tao Qin, and Tie-Yan Liu, "Token-level ensemble distillation for grapheme-to-phoneme conversion," *arXiv preprint arXiv:1904.03446*, 2019.

[6] Daju Gou and Wanbo Luo, "Processing of polyphone character in chinese tts system," *Chinese Information*, vol. 1, pp. 33–36.

[7] Fang-zhou Liu, Qin Shi, and J Tao, "Maximum entropy based homograph disambiguation," *NCMMSC2007*, 2007.

[8] Fan Ming Hu Guoping Wang Renhua, "Multi-level polyphone disambiguation for mandarin grapheme-phoneme conversion," *Computer Engineering and Applications*, , no. 2, pp. 49, 2006.

[9] Fangzhou Liu and You Zhou, "Polyphone disambiguation based on tree-guided tbl," *Jisuanji Gongcheng yu Yingyong(Computer Engineering and Applications)*, vol. 47, no. 12, pp. 137–140, 2011.

[10] Changhao Shan, Lei Xie, and Kaisheng Yao, "A bi-directional lstm approach for polyphone disambiguation in mandarin chinese," in *2016 10th International Symposium on Chinese Spoken Language Processing (ISCSLP)*. IEEE, 2016, pp. 1–5.

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

[12] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever, "Improving language understanding by generative pre-training," *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*, 2018.

[13] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu, "Mass: Masked sequence to sequence pre-training for language generation," *arXiv preprint arXiv:1905.02450*, 2019.

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

[15] Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin, "Distilling task-specific knowledge from bert into simple neural networks," *arXiv preprint arXiv:1903.12136*, 2019.

[16] Hong Zhang, JiangSheng Yu, WeiDong Zhan, and Shi-Wen Yu, "Disambiguation of chinese polyphonic characters," in *The First International Workshop on Multi-Media Annotation (MMA2001)*, 2001, vol. 1, pp. 30–1.

[17] Fangzhou Liu, Qin Shi, and Jianhua Tao, "Tree-guided transformation-based homograph disambiguation in mandarin tts system," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2008, pp. 4657–4660.

[18] Zexin Cai, Yaogen Yang, Chuxiong Zhang, Xiaoyi Qin, and Ming Li, "Polyphone Disambiguation for Mandarin Chinese Using Conditional Neural Network with Multi-level Embedding Features," *arXiv e-prints*, p. arXiv:1907.01749, Jul 2019.

[19] 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.

[20] Joseph Fisher and Andreas Vlachos, "Merge and Label: A novel neural network architecture for nested NER," *arXiv e-prints*, p. arXiv:1907.00464, Jun 2019.

[21] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, pp. 535–541.

[22] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[23] Tommaso Furlanello, Zachary C Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar, "Born again neural networks," *arXiv preprint arXiv:1805.04770*, 2018.

[24] Rohan Anil, Gabriel Pereyra, Alexandre Passos, Robert Ormandi, George E Dahl, and Geoffrey E Hinton, "Large scale distributed neural network training through online distillation," *arXiv preprint arXiv:1804.03235*, 2018.

[25] Chenglin Yang, Lingxi Xie, Siyuan Qiao, and Alan Yuille, "Knowledge distillation in generations: More tolerant teachers educate better students," *arXiv preprint arXiv:1805.05551*, 2018.

[26] Yoon Kim and Alexander M Rush, "Sequence-level knowledge distillation," *arXiv preprint arXiv:1606.07947*, 2016.

[27] Markus Freitag, Yaser Al-Onaizan, and Baskaran Sankaran, "Ensemble distillation for neural machine translation," *arXiv preprint arXiv:1702.01802*, 2017.

[28] Xu Tan, Yi Ren, Di He, Tao Qin, and Tie-Yan Liu, "Multilingual neural machine translation with knowledge distillation," in *International Conference on Learning Representations*, 2019.

[29] Zehao Huang and Naiyan Wang, "Like what you like: Knowledge distill via neuron selectivity transfer," *arXiv preprint arXiv:1707.01219*, 2017.