Xu YG, Qiu XP, Zhou LG *et al.* Improving BERT fine-tuning via self-ensemble and self-distillation. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 38(4): 853-866 July 2023. DOI: 10.1007/s11390-021-1119-0

Improving BERT Fine-Tuning via Self-Ensemble and Self-Distillation

Yi-Ge Xu[1, 2] (许一格), *Student Member, CCF*, Xi-Peng Qiu[1, 2, *] (邱锡鹏), *Member, CCF*
Li-Gao Zhou[3] (周浬皋), and Xuan-Jing Huang[1, 2] (黄萱菁), *Member, CCF*

[1] *School of Computer Science, Fudan University, Shanghai 200433, China*

[2] *Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, Shanghai 200433, China*

[3] *Huawei Technologies Co., Ltd., Hangzhou 310052, China*

E-mail: ygxu18@fudan.edu.cn; xpqiu@fudan.edu.cn; zhouligao@huawei.com; xjhuang@fudan.edu.cn

Received October 28, 2020; accepted June 18, 2021.

Abstract Fine-tuning pre-trained language models like BERT have become an effective way in natural language processing (NLP) and yield state-of-the-art results on many downstream tasks. Recent studies on adapting BERT to new tasks mainly focus on modifying the model structure, re-designing the pre-training tasks, and leveraging external data and knowledge. The fine-tuning strategy itself has yet to be fully explored. In this paper, we improve the fine-tuning of BERT with two effective mechanisms: self-ensemble and self-distillation. The self-ensemble mechanism utilizes the checkpoints from an experience pool to integrate the teacher model. In order to transfer knowledge from the teacher model to the student model efficiently, we further use knowledge distillation, which is called self-distillation because the distillation comes from the model itself through the time dimension. Experiments on the GLUE benchmark and the Text Classification benchmark show that our proposed approach can significantly improve the adaption of BERT without any external data or knowledge. We conduct exhaustive experiments to investigate the efficiency of the self-ensemble and self-distillation mechanisms, and our proposed approach achieves a new state-of-the-art result on the SNLI dataset.

Keywords BERT, deep learning, fine-tuning, natural language processing (NLP), pre-training model

1 Introduction

The pre-trained language models including BERT[1] and its variants (XLNet[2] and RoBERTa[3]) have been proven beneficial for many natural language processing (NLP) tasks, such as text classification, question answering[4] and natural language inference[5]. These pre-trained models have learned general-purpose language representations on a large amount of unlabeled data. Therefore, adapting these models to the downstream tasks can bring a good initialization and avoid training from scratch. There are two common ways to utilize these pre-trained models on downstream tasks: feature extraction (where the pre-trained parameters are frozen), and fine-tuning (where the pre-trained parameters are unfrozen and fine-tuned)[6]. Although both ways can significantly improve the performance of most downstream tasks, the fine-tuning way usually achieves better results than the feature extraction way[7]. Thus it is worth paying attention to find a good fine-tuning strategy.

As a widely-studied pre-trained language model, the potential of BERT can be further boosted by modifying the model structure[8, 9] and re-designing pre-training objectives[2, 3, 10, 11], data augmentation[12] and multi-stage transfer[13]. However, the fine-tuning strategy itself has yet to be fully explored.

In this paper, we investigate how to maximize the utilization of BERT by a better fine-tuning strategy without utilizing external data or knowledge. BERT is usually fine-tuned by using the stochastic gradient descent (SGD) method. In practice, the performance of fine-tuning BERT is often sensitive to different random seeds and the order of training data, especial-

Regular Paper

This work was supported by the National Key Research and Development Program of China under Grant No. 2020AAA0106700 and the National Natural Science Foundation of China under Grant No. 62022027.

*Corresponding Author

ly when the last training sample is noise. To alleviate this, the ensemble method[14] is widely-used to combine several fine-tuned based models since it can reduce overfitting and improve model generalization. The ensemble BERT usually achieves superior performance to the single BERT model. However, the main disadvantages of the ensemble method are its model size and training cost. The ensemble model needs to keep multiple fine-tuned BERTs and has low computation efficiency and high storage cost.

As illustrated in Fig.1, we improve the fine-tuning approach of BERT by introducing two mechanisms: self-ensemble and self-distillation.

Self-Ensemble. Motivated by the success of widely-used ensemble models, we firstly propose a self-ensemble method, in which the base models are the previous checkpoints within a single training process[15]. Then we compute the output through a majority vote. To further reduce the model complexity of the ensemble model, we use a more efficient ensemble method, which combines several base models with parameter averaging rather than keep several base models. In summary, in order to compute the output logit, there are two self-ensemble methods: one for computing the average of the result, and the other for computing the average of parameters. Furthermore, to break the temporal correlations by mixing more and less recent checkpoints, we introduce an experience pool[16] to store the past checkpoints. In each step, some models in the experience pool are sampled to establish the teacher model via self-ensemble.

Self-Distillation. We further use knowledge distillation[17] to improve fine-tuning efficiency. In this paper, time step t indicates that the model parameters have been optimized by t times, and time step 0 indicates the initial parameters. At each time step in training, the current BERT model (called student model) is learned with two teachers: the gold labels and self-ensemble model (called teacher model). With the help of the teacher model, the student model is more robust and accurate. Moreover, a better student model further leads to a better teacher model. A similar idea is also used in semi-supervised learning, such as temporal ensembling[18] and mean teacher[19]. Different from them, our proposed self-distillation aims to optimize the student model without external data. More recently, ODC[20] and FastBERT[21] also introduce self-distillation.

Different from previous self-distillation mechanisms that distill knowledge from the following three perspectives: previous best checkpoints[20], outputs of other transformer layers[21], and the L2 distance to the prediction with augmented data[19], our self-distillation method distills knowledge from an ensemble of past checkpoints of the student model.

In this paper, we propose a simple but effective fine-tuning approach containing two mechanisms: self-ensemble and self-distillation. The self-ensemble mechanism includes two methods of parameter averaging and logits voting respectively. The self-distillation mechanism include one method that distills knowledge from the teacher model constructed by the

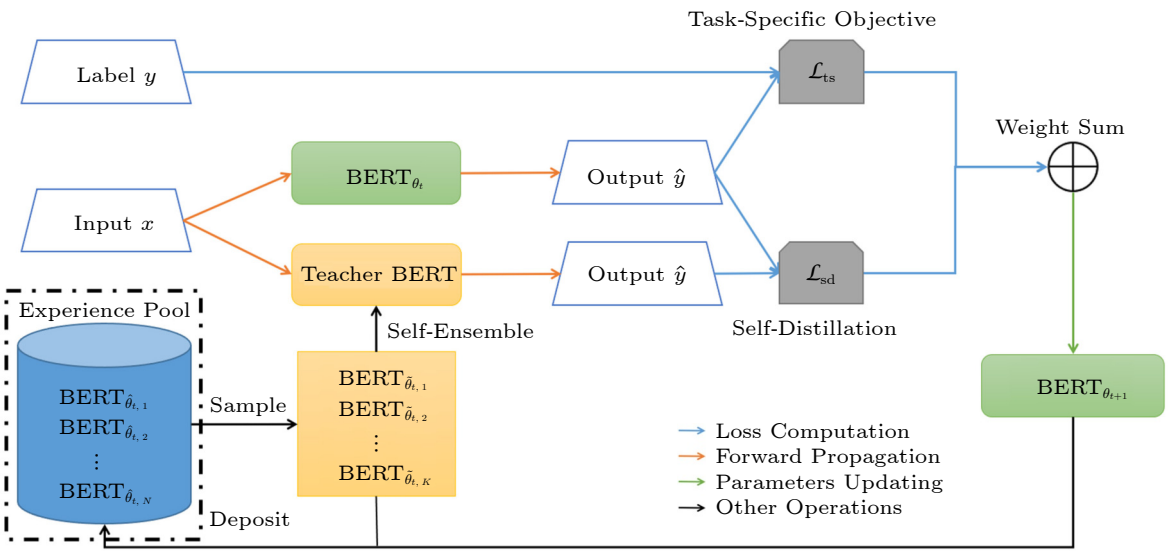


Fig.1. Illustration of our proposed fine-tuning approach. θ_t is the current parameters which have been optimized by t times. (x, y) is the input training sample. $\mathcal{L}_{\text{sd}}(\cdot)$ and $\mathcal{L}_{\text{ts}}(\cdot)$ denote the self distillation loss and the task-specific loss, respectively. N and K are the size of the experience pool and the size of the teacher model respectively.

self-ensemble mechanism. We evaluate our fine-tuning approach on the GLUE benchmark and the classification benchmark. The experiments on the GLUE benchmark[22] show our proposed approach has an average of 0.9-point performance boost compared with vanilla BERT fine-tuning. The experiments on the classification benchmark show our proposed approach has the average of relative change of 6.26%. To further analyze our approach, we conduct exhaustive experiments including analyzing the training stability, the training curves, and the difference between the two self-ensemble methods.

The contributions of this paper are as follows.

• We show the potential of BERT can be further stimulated by a better fine-tuning approach without leveraging external knowledge or data.

• We propose two self-ensemble methods: one for computing the average of the result, and the other for computing the average of parameters. The methods can improve BERT without significantly decreasing the training efficiency.

• We propose a self-distillation method with experience replay. This method can utilize the distillation mechanism to transfer knowledge from teacher models constructed by the self-ensemble mechanism.

This paper is organized as follows: Section 2 reviews some related work, Section 3 introduces our proposed fine-tuning approach, Section 4 shows the experimental setup and results, and Section 5 concludes the paper.

2 Related Work

We briefly review three kinds of related work: pre-trained language models, knowledge distillation for NLP applications, and self-distillation.

2.1 Pre-Trained Language Models

It has become a new paradigm for NLP that first pre-train language models on a large amount of unlabeled data and then fine-tune the parameters in downstream tasks[6]. It also makes a breakthrough in many NLP tasks. Most recent pre-trained language models (e.g., BERT[1], XLNet[2] and RoBERTa[3]) are built with Transformer architecture[23].

As a wide-used model, BERT is pre-trained on the masked language model (MLM) task and next sentence prediction (NSP) task via a large cross-domain unlabeled corpus. When fine-tuning on downstream tasks, BERT takes an input of a sequence of no more than 512 tokens and outputs the representation of the sequence. The sequence has one segment for single-sentence tasks or two for pairwise-sentence tasks. A special token [CLS] is added before segments which contain the sequence representations. Another special token [SEP] is used for separating segments.

2.2 Knowledge Distillation for NLP Applications

Since the pre-trained language models usually have an extremely large number of parameters, fine-tuning them on a downstream task usually needs a high computation cost, which is difficult to deploy on the resource-restricted devices. [17] proposes knowledge distillation with a teacher-student architecture to transfer the knowledge from a large teacher model to a small student model by reproducing the behaviors of the teacher model. Following the teacher-student architecture, recent studies[24–27] have designed special objective functions to distill knowledge from pre-trained language models.

2.3 Self-Distillation

Generally speaking, the teacher model is usually well-trained and fixed in the processing of knowledge distillation and has more parameters than the student one, and the student model is trained with a objective function computed by the distillation mechanism. Moreover, despite distilling from the teacher model, the student one also can distill knowledge by itself throughout an online knowledge distillation.

On computer vision (CV), a similar idea is also used in semi-supervised learning, such as temporal ensembling[18] and mean teacher[19]. During the semi-supervised learning, the input data is augmented by a disturbed δ. The objective of the mean teacher is to minimize the L2 distance between the prediction of the teacher model and that of the student model.

On NLP, online distillation from the best checkpoint (ODC)[20] is applied on neural machine translation tasks. The teacher model of ODC is updated when the checkpoint is the best on the validation data, and it is used to lead the training of the student model when the validation performance declines. Moreover, FastBERT[21] contains a self-distillation loss function that distills knowledge from the outputs of the last layer to the outputs of other layers.

2.4 Neural Network Ensemble

According to previous surveys[14, 28], the neural network ensemble method can be separated into three parts. The first part is about the individual network generation method, including applying diverse input data, redesigning ensemble objective function, and selecting neural networks. The second part is about the conclusion generation method, including the absolute majority vote and the relative majority vote. The last part is about the module ensemble method, including fusing garnular computing (GrC).

3 Methodology of Fine-Tuning BERT

The fine-tuning of BERT usually aims to minimize the cross-entropy loss on a specific task with the stochastic gradient descent method. Due to the stochastic nature, the performance of fine-tuning is often affected by the random orderings of the training data, especially when the last training sample is noise.

Our fine-tuning method is motivated by the ensemble method and knowledge distillation. It consists of two models: a student model is the fine-tuning BERT, and a teacher model is a self-ensemble of several student models sampled from the experience pool. At each time step, we further distill the knowledge of the teacher model to the student model.

3.1 Fine-Tuning Vanilla BERT

BERT can deal with different natural language tasks with task-specific output layers. For the GLUE benchmark, BERT takes the final hidden state $\boldsymbol{h}$ of the first token [CLS] as the representation of the input sentence or sentence-pair. A simple softmax classifier is added to the top of BERT to predict the probability of label y:

$$p(y|\boldsymbol{h}) = \text{softmax}(\boldsymbol{W}\boldsymbol{h}),$$

where $\boldsymbol{W}$ is the task-specific parameter matrix. A task-specific loss $\mathcal{L}_{\text{ts}}(\cdot)$ is used to fine-tune BERT as well as $\boldsymbol{W}$ jointly:

$$\mathcal{L}_{\text{ts}}(\cdot) = \begin{cases} CE(\hat{y}, y), & \text{for classification tasks,} \\ MSE(\hat{y}, y), & \text{for regression tasks,} \end{cases} \quad (1)$$

where $\hat{y} = p(y|\boldsymbol{h})$ is predicted by the BERT model, while $CE(\cdot)$ and $MSE(\cdot)$ indicate cross-entropy loss and mean square error respectively.

3.2 Self-Ensemble

The basic idea of self-ensemble is inspired by the famous proverb "look before you leap". On vanilla BERT fine-tuning, the parameters are optimized by an objective function which only relies on current parameters θ_t as well as the training sample (x, y). Previous work shows that experience replay benefits to reinforcement learning[16, 29, 30]. Motivated by this, we propose the self-ensemble mechanism throughout an experience pool.

Let θ_t denote parameters when fine-tuning BERT at time step t. The experience pool is defined as:

$$\hat{\Theta}_t = \{\hat{\theta}_{t,1}, \hat{\theta}_{t,2}, \ldots, \hat{\theta}_{t,N}\},$$

where N indicates the size of the experience pool, and $\hat{\theta}_{t,i} \in \{\theta_0, \theta_1, \ldots, \theta_t\}$ for all $i \in [1, N]$.

Then, we randomly sample models from the experience pool to establish the teacher model:

$$\tilde{\Theta}_t = \{\tilde{\theta}_{t,1}, \tilde{\theta}_{t,2}, \ldots, \tilde{\theta}_{t,K}\}, \quad (2)$$

where $K \in [0, N]$ indicates the teacher size, and $\tilde{\theta}_{t,i} \in \hat{\Theta}_t$ for all $i \in [1, K]$. Specially, $K = 0$ indicates the vanilla BERT fine-tuning mentioned in Subsection 3.1.

In the first N steps, θ_t are filled into the experience pool. After that, one of the teacher models will be removed from the experience pool when the parameters have been optimized. In order to keep the diversity of the experience pool as much as possible, instead of removing the oldest checkpoint, the removed model is selected randomly:

$$\hat{\Theta}_{t+1} = \left(\hat{\Theta}_t - \{\tilde{\theta}_{t,j}\}\right) \cup \{\theta_{t+1}\},$$

where $\tilde{\theta}_{t,j}$ indicates the removed model.

In the training phase, each of the checkpoints θ_i will participate the composition of the teacher model at least once and with the expectation of K times.

As mentioned above, K models are selected as $\tilde{\Theta}_t$, which integrates the teacher model. As shown in Fig.2, there are two different methods to establish the teacher model: parameter averaging and logits voting. Usually, parameter averaging has better computational and memory efficiency than logits voting.

3.3 Self-Distillation with Experience Replay

Although the self-ensemble mechanism contains an experience pool, it cannot transfer knowledge from the teacher model to the student model directly.

Fig.2. Two methods to compute the output of the teacher model. (a) Parameter averaging. (b) Logits voting.

Thus, we propose self-distillation mechanism to transfer the "dark knowledge"[17].

Previous distillation models[24, 25, 31, 32] mainly focus on a two-stage training: 1) training the teacher model, 2) fixing the teacher model and training the student model. This training procedure leads to the teacher model not being improved during the distillation procedure. But, our proposed self-ensemble mechanism utilizes the experience pool's checkpoints to build the dynamic teacher model. With the help of the teacher model, the student model will become more robust and more accurate. With a better student model, the teacher model at the future time step is better either. In the training phase, a self-distillation loss $\mathcal{L}_{\text{sd}}(\cdot)$ is added to the objective function:

$$\mathcal{L}(\cdot) = \mathcal{L}_{\text{ts}}(\cdot) + \lambda \mathcal{L}_{\text{sd}}(\cdot),$$

where $\mathcal{L}_{\text{ts}}(\cdot)$ is defined at (1), and λ is the weight of the self-distillation loss. The model parameters θ_t are optimized by $\mathcal{L}(\cdot)$.

Following the two methods establishing the teacher model, there are two kinds of self-distillation loss: self-distillation averaged, and self-distillation voted.

3.3.1 *Self-Distillation Averaged (SDA)*

We first denote a fine-tuning method BERT_{SDA}, in which the teacher model is self-ensemble BERT with parameter averaging.

Previous work has shown that averaging model weights over training steps tends to produce a more accurate model than using the final weights directly[15]. Following this, the teacher model with parameter averaging can be computed by:

$$\bar{\theta}_t = \frac{1}{K} \sum_{k=1}^{K} \tilde{\theta}_{t,k},$$

where K and $\tilde{\theta}_{t,k}$ are defined in (2).

Similar to other self-distillation models[20, 21], the self-distillation loss of BERT_{SDA} does not rely on label y, but only relies on input x. We use mean squared error loss to estimate the logit-divergence between the student model and the teacher model:

$$\mathcal{L}_{\text{sd}}(x) = MSE\Big(BERT_{\vartheta_t}(x), BERT_{\bar{\theta}_t}(x)\Big),$$

where MSE denotes the mean squared error, and $BERT_{\theta}(\cdot)$ denotes the logits outputted by the BERT models with parameters θ.

More specially, we use a moving average to record the average of parameters from θ_0 to θ_{t-1}, which means the size of experience pool N and the size of the teacher model K both keep moving with t. For consistency, we define this case as $\text{BERT}_{\text{SDA}(K=t)}$.

3.3.2 *Self-Distillation Voted (SDV)*

As a comparison, we also propose an alternative self-distillation method BERT_{SDV} by establishing the teacher model through logits voting.

Different from parameter averaging, logits voting needs to compute logits on each teacher model and then sums the logits up. The self-distillation loss of the BERT_{SDV} method can be defined as:

$$\mathcal{L}_{\text{sd}}(x) = MSE\left(BERT_{\theta_t}(x), \frac{1}{K}\sum_{k=1}^{K} BERT_{\tilde{\theta}_{t,k}}(x)\right).$$

Since the teacher model aggregates the information of student models after every time step, it is usually more robust than a single student model without self-distillation. Moreover, a better student model further leads to a better teacher model. Generally, the training efficiency of BERT_{SDV} is lower than that of BERT_{SDA} since BERT_{SDV} needs to process the input with K teacher models. Specially, BERT_{SDV} is the same as BERT_{SDA} when $K = 1$.

4 Experiments

We improve BERT fine-tuning via self-ensemble and self-distillation. The vanilla fine-tuning method of BERT is used as our baseline. Then we evaluate our proposed fine-tuning method on the GLUE benchmark[22] and classification benchmarks to demon-

strate the feasibility of our self-distillation model.

4.1 Datasets

Our proposed method is evaluated on the GLUE benchmark[22] and classification benchmarks. The statistics of the benchmarks are shown in Table 1 and Table 2 respectively. We use "#" to indicate "the number of".

4.1.1 GLUE Benchmark

CoLA. The Corpus of Linguistic Acceptability (CoLA) is a binary classification task to predict whether the given English sentence is "linguistically acceptable" or not[33].

SST-2. The Stanford Sentiment Treebank (SST-2) is a corpus extracted from movie reviews with human annotations of their sentiment[34]. The task aims to predict whether the reviews are positive or negative.

MRPC. The Microsoft Research Paraphrase Corpus (MRPC)[35] consists of sentence pairs automatically extracted from online news sources. The task aims to predict the human annotations for whether the sentences in the pair are semantically equivalent.

STS-B. The Semantic Textual Similarity Benchmark (STS-B)[36] is a collection of sentence pairs collected from news headlines, video, image captions, and other resources. The data is annotated through a continuous score from 0 to 5, which denotes how similar the two sentences are. This task is evaluated by Pearson correlation (P. corr) and Spearman correlation (S. corr).

QQP. The Quora Question Pairs (QQP)① is a binary classification task aiming to predict whether two questions asked on Quora are semantically equivalent.

MNLI. The Multi-Genre Natural Language Inference (MNLI)[37] corpus is a crowdsourced entailment classification task with about 433k sentence pairs. It has two development sets and two test sets: matched (MNLI-m) and mismatched (MNLI-mm).

QNLI. The Question Natural Language Inference (QNLI)[22] is a binary classification task built from the Stanford Question Answering Dataset[4]. The task aims to predict whether the question-sentence pair contains the correct answer or not.

RTE. The Recognizing Textual Entailment (RTE)[22] dataset is collected from a series of annual challenges on textual entailment.

4.1.2 Classification Benchmark

IMDb. IMDb[39] is a binary sentiment analysis dataset from the Internet Movie Database. The dataset has 25 000 training examples and 25 000 validation examples. The task is to predict whether the review text is positive or negative.

AG's News. AG's corpus[40] of the news articles on

Table 1. Summary Statistics of GLUE Benchmark

Dataset	#Labels	#Train Samples	#Development Samples	#Test Samples	Metric
CoLA[33]	2	8 551	1 043	1 063	Matthews correlation coefficient[38]
SST-2[34]	2	67 349	872	1 821	Accuracy
MRPC[35]	2	3 668	408	1 725	Accuracy/F_1
STS-B[36]	1	5 749	1 500	1 379	Pearson/Spearman correlation
QQP①	2	363 849	40 430	390 965	Accuracy/F_1
MNLI-m[37]	3	392 702	9 815	9 796	Accuracy
MNLI-mm[37]	3	392 702	9 832	9 847	Accuracy
QNLI[22]	2	104 743	5 463	5 463	Accuracy
RTE[22]	2	2 490	277	3 000	Accuracy

Table 2. Summary Statistics of Six Widely-Studied Text Classification and Natural Language Inference (NLI) Datasets

Type	Dataset	#Labels	#Train Samples	#Development Samples	#Test Samples
Text classification	IMDb[39]	2	25 000	0	25 000
	AG's News[40]	4	120 000	0	7 600
	DBPedia[40]	14	560 000	0	70 000
	Yelp Polarity[40]	2	560 000	0	38 000
	Yelp Full[40]	5	650 000	0	50 000
NLI	SNLI[5]	3	549 367	9 842	9 824

①http://static.hongbozhang.me/doc/STAT_441_Report.pdf, Jun. 2021.

the web contains 496 835 categorized news articles. The four largest classes from this corpus with only the title and description fields are chosen to construct the AG's News dataset.

DBPedia. DBPedia[40] is a crowd-sourced community effort that includes structured information from Wikipedia. The DBPedia dataset is constructed by picking 14 non-overlapping classes from DBPedia 2014 with only the title and abstract of each Wikipedia article.

Yelp. The Yelp dataset is obtained from the Yelp Dataset Challenge in 2015, built by [40]. There are two classification tasks in this dataset: Yelp Full and Yelp Polarity. Yelp Full predicts the full number of stars (1 to 5) which are given by users, and the other predicts a polarity label that is positive or negative.

SNLI. The Stanford Natural Language Inference Corpus[5] is a collection of 570k human-written English sentence pairs manually labeled for balanced classification with the labels on entailment, contradiction, or neutral.

4.2 Implementation Details

The selection of hyperparameters N, K, and λ which are defined in Section 3, can be seen in Subsection 4.3. The other hyperparameters of our proposed methods are the same as those in the official BERT[1].

We use AdamW optimizer with the warm-up proportion of 0.1, base learning rate for the BERT encoder of 2×10^{-5}, base learning rate for the softmax layer of 1×10^{-3}, and dropout probability of 0.1. For sequences of over 512 tokens, we truncate them and choose head 512 as the model input. We fine-tune all models on one RTX 2080Ti GPU. Due to the limitation of GPU memory, the batch size is different from 4 (for classification tasks with long sequence length) to 16 (for GLUE benchmark with short sequence length) and the gradient accumulation step is 8.

Taking IMDb as an example, BERT_{SDA} with $N = K = 5$ spends about 1.5x training time compared with the vanilla fine-tuning, while it is about 2.8x–3x training times in the case of BERT_{SDV} with $N = K = 5$.

4.3 Model Selection

As shown in Subsection 3.2 and Subsection 3.3, there are three main hyperparameters in our fine-tuning methods BERT_{SDA} and BERT_{SDV}: size of the experience pool N, size of the teacher model K, and self-distillation weight λ.

In this subsection, we mainly choose four representative tasks to evaluate the effects of our hyperparameters: a task for single-sentence classification (SST-2), a task for pairwise-sentence classification (MRPC), a task for text-similarity regression (STS-B), and a task for relevance ranking (QNLI). In the tables of this paper, the best results are in bold.

4.3.1 *Size of Experience Pool N*

In this subsection, we will explore the effects of N. For convenience, we firstly set $K = 3$ and $\lambda = 0.1$. Then we change the size of experience pool N. Results are shown in Table 3.

According to the experimental result, some tasks are not sensitive to the value of N, while the other tasks are sensitive. Similar observations can be found on both BERT_{SDA} and BERT_{SDV}. For sensitive tasks such as SST-2 and MRPC, our fine-tuning method usually gets a good result when $N = 5$. Therefore, we set $N = 5$ in the following experiments.

4.3.2 *Size of Teacher Model K*

In this subsection, we choose different sizes of the teacher model and evaluate our model in four tasks. Following Subsection 4.3.1, we set $N = 5$ and $\lambda = 0.1$. Results are shown in Table 4.

From the experimental results, the size of the teacher model is sensitive to datasets. Thus, we select the best K for each task in the following experiment.

Table 3. Effects (%) of N on the Development Set of Four Representative Tasks

Task	Metric	BERT_{SDV}					BERT_{SDA}				
		$N = 3$	$N = 4$	$N = 5$	$N = 8$	$N = 10$	$N = 3$	$N = 4$	$N = 5$	$N = 8$	$N = 10$
SST-2	Accuracy	93.3	93.2	93.5	93.0	92.8	93.3	92.8	**93.5**	92.9	92.8
MRPC	Accuracy/F_1	86.3	87.5	**89.0**	88.2	86.5	86.2	86.3	88.2	86.5	87.3
STS-B	P. corr	90.0	90.0	**90.0**	90.0	90.0	89.9	89.9	90.0	89.9	90.0
	S. corr	89.7	89.7	**89.7**	89.6	89.7	89.5	89.6	89.6	89.5	89.6
QNLI	Accuracy	91.2	91.4	91.3	91.2	91.2	91.5	91.3	**91.8**	91.5	91.4

Table 4. Effects (%) of K on the Development Set of Four Representative Tasks

Task	Metric	BERT_{SDV}					BERT_{SDA}			
		$K=1$	$K=2$	$K=3$	$K=4$	$K=5$	$K=2$	$K=3$	$K=4$	$K=5$
SST-2	Accuracy	92.9	93.5	**93.5**	93.0	93.2	93.1	93.5	92.9	93.1
MRPC	Accuracy/F_1	87.0	88.1	**89.0**	87.8	86.7	86.8	87.3	87.2	87.2
STS-B	P. corr	90.1	90.0	90.0	90.0	**90.2**	90.0	90.0	90.0	90.1
	S. corr	89.8	89.7	89.7	89.7	**89.8**	89.6	89.6	89.6	89.7
QNLI	Accuracy	91.5	91.2	91.3	91.4	91.4	91.7	**91.8**	91.5	91.7

4.3.3 *Self-Distillation Weight λ*

In this part, we conduct experiments to explore the effects of λ. Following Subsection 4.3.1 andSubsection 4.3.2, we set $N=5$ and choose different K.

As shown in Fig.3, $\lambda \in [0.1, 0.3]$ has better results in STS-B than the other ranges, while $\lambda \in [1.0, 1.5]$ performs better than the other ranges in IMDb. Our observation on loss curve (see Subsection 4.5.2) shows that the norms between task-specific loss and self distillation loss should apply within a suitable ratio. Therefore, λ should be set in $[1.0, 1.5]$ for large-scaled tasks, such as IMDb, SST-2, and so on. In contrast, λ should be set in $[0.1, 0.3]$ for the task with limited training examples, such as MRPC, STS-B, and so on.

4.4 Model Performance

We explored the selection of hyperparameters in Subsection 4.3. In this subsection, we will evaluate our proposed fine-tuning method for the BERT and RoBERTa models on the GLUE benchmark and the classification benchmarks.

4.4.1 *Effects on Fine-Tuning BERT-Base*

In this subsection, we use 12-layer Transformer encoders (BERT-Base) as our mainly component and evaluate our methods on the GLUE benchmark.

As shown in Table 5 and Table 6, where Mcc means Matthew's correlation coefficient and Acc means accuracy, our self-ensemble and self-distillation method improve fine-tuned models on the GLUE benchmark. We firstly re-implement fine-tuning vanilla BERT and submit it to the GLUE server as our baseline. Compared with the baseline, our self-ensemble and self-distillation method has a 0.9-point performance boost on the test set of the GLUE benchmark.

4.4.2 *Effects on Fine-Tuning BERT-Large and RoBERTa-Large*

We also investigate whether self-distillation has similar findings for the $\text{BERT}_{\text{LARGE}}$ (BERT-L) and $\text{RoBERTa}_{\text{LARGE}}$ (RoBERTa-L) model, containing 24 Transformer layers. For convenience, we set two different sizes K of the teacher model for comparison. For IMDb and AG's News, we report test error rate (%). For SNLI, we report accuracy (%). XLNet has

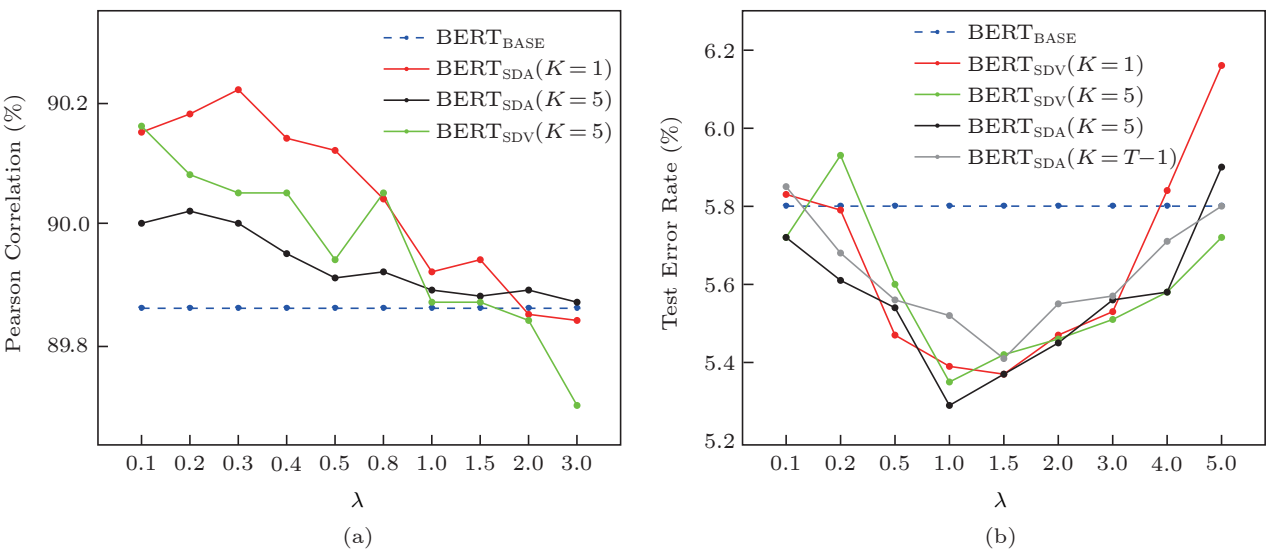


Fig.3. Comparison of different λ on: (a) STS-B (development set) and (b) IMDb (test set). T: the total number of iterations.

Table 5. Model Comparison (%) on the Development Set of the GLUE Benchmark

Model	CoLA (Mcc)	SST-2 (Acc)	MRPC (Acc/F_1)	STS-B (P./S. Corr)	QQP (Acc/F_1)	MNLI-m/mm (Acc)	QNLI (Acc)	RTE (Acc)
$\text{BERT}_{\text{BASE}}$[1]	-	92.7	86.7/-	-	-	84.4/-	88.4	-
$\text{BERT}_{\text{BASE}}$-ReImp	56.7	92.7	85.7/-	89.9/89.6	91.2/88.2	84.4/84.2	91.0	67.1
BERT_{SDA}(ours)	**60.7**	93.5	**89.0**/-	**90.2**/**89.8**	91.4/**88.4**	84.8/**85.2**	**91.8**	71.8
BERT_{SDV}(ours)	60.5	**93.5**	88.2/-	90.0/89.7	**91.4**/88.4	**85.1**/85.1	91.5	**72.2**

Note: "-" means not reported in [1]. "ReImp" indicates our implementation. We report results on the development set with the best hyperparameters.

Table 6. Model Comparison (%) on the Test Set of the GLUE Benchmark

Model	CoLA (Mcc)	SST-2 (Acc)	MRPC (Acc/F_1)	STS-B (P./S. Corr)	QQP (Acc/F_1)	MNLI-m/mm (Acc)	QNLI (Acc)	RTE (Acc)	Avg. Score
$\text{BERT}_{\text{BASE}}$[1]	52.1	93.5	88.9/84.8	87.1/85.8	71.2/89.2	84.6/83.4	90.5	66.4	79.7
$\text{BERT}_{\text{BASE}}$-ReImp	52.2	93.4	88.3/84.8	86.7/85.6	71.0/89.2	84.3/83.4	90.5	66.5	79.6
BERT_{SDA}(ours)	**53.1**	94.4	**88.7**/**84.5**	**87.0**/**86.0**	72.4/89.6	85.0/84.3	91.3	68.8	**80.6**
BERT_{SDV}(ours)	52.6	**94.6**	88.4/84.4	86.9/85.7	**72.5**/**89.7**	**85.3**/**84.3**	**91.4**	**68.9**	80.5

the state-of-the-art result on text classification tasks. MT-DNN fine-tunes BERT with multi-task learning. CA-MTL fine-tunes RoBERTa with multi-task learning. In Table 7, self-distillation also gets a significant gain while fine-tuning BERT-L. On two text classification tasks, $\text{BERT-L}_{\text{SDA}}$ and $\text{RoBERTa-L}_{\text{SDA}}$ give better results than self-distillation-voted models. The average improvement (Avg. Δ) of $\text{BERT-L}_{\text{SDA}}$ and $\text{RoBERTa-L}_{\text{SDA}}$ is 7.02% and 5.81%, respectively. For the NLI task, $\text{BERT-L}_{\text{SDV}}$ and $\text{RoBERTa-L}_{\text{SDV}}$ give better results than self-distillation-average models. The average improvement of $\text{BERT-L}_{\text{SDV}}$ and $\text{RoBERTa-L}_{\text{SDV}}$ are 6.59% and 9.76%, respectively.

Moreover, although our self-distillation mechanism does not leverage the external data or knowledge, it also gives a comparable performance of MT-DNN[11] on BERT-L and outperforms CA-MTL[41] on RoBERTa-L. MT-DNN and CA-MTL fine-tune BERT and RoBERTa under the multi-task learning framework respectively. In SNLI, MT-DNN achieves the best result except the RoBERTa models, and CA-MTL is the previous state-of-the-art model.

4.5 Model Analysis

In this subsection, we will briefly analyze our proposed fine-tuning method in two perspectives: training stability and convergence curves.

4.5.1 Training Stability

Fine-tuning a pre-trained model on downstream tasks prevents training a model from scratch, which usually requires a high computation power. Meanwhile, distinct random seeds can lead to substantially different results when fine-tuning BERT even with the same hyperparameters. In our fine-tuning method, a self-distillation loss function is added to the objective function. This function can be regarded as a constraint on regularization. Thus, we assume that the parameter averaging and logits voting can increase

Table 7. Comparison of Different 24-Layer Models

Model	Test Error Rate (%)		Avg. Δ (%)	Accuracy of SNLI (%)	Δ (%)
	IMDb	AG's News			
XLNet[2]	3.79	4.49	/	/	/
MT-DNN[11]	/	/	/	91.6	/
CA-MTL[41]	/	/	/	92.1	/
BERT-L (our implementation)	4.98	5.45	-	90.9	-
RoBERTa-L (our implementation)	3.88	5.33	-	91.8	-
$\text{BERT-L}_{\text{SDV}}$	4.66	5.21	5.62	**91.5**	**6.59**
$\text{BERT-L}_{\text{SDA}}$	**4.58**	**5.15**	**7.02**	91.4	5.49
$\text{RoBERTa-L}_{\text{SDV}}$	3.58	5.03	5.62	**92.6**	**9.76**
$\text{RoBERTa-L}_{\text{SDA}}$	**3.48**	**5.02**	**5.81**	92.5	8.54

Note: "/" indicates "not reported in the original paper". "-" means the baseline.

the ability of generalization and reduce the variance of model performance. In this subsection, we conduct experiments to explore the effect of data order during fine-tuning and prove our assumption.

This experiment is conducted with a set of data order seeds. A data order can be regarded as a sample of the set of permutations of the training data. In this subsection, the same θ_0 is used to initialize the weight matrices of all models, and the data orders are different. We run 10 times for each fine-tuning strategy and record the results as shown in Table 8.

Table 8. Statistics of Evaluation Metrics in STS-B Within 10 Runs

Model	Metric	Max	Min	Avg.	Std.
$\text{BERT}_{\text{BASE}}$	P. Corr	90.05	89.66	89.86	1.537×10^{-1}
	S. Corr	89.64	89.30	89.48	1.249×10^{-1}
$\text{BERT}_{\text{SDA}}(K=1)$	P. Corr	90.15	89.79	89.98	9.910×10^{-2}
	S. Corr	89.79	89.48	89.62	8.710×10^{-2}
$\text{BERT}_{\text{SDA}}(K=5)$	P. Corr	90.13	89.92	90.03	6.870×10^{-2}
	S. Corr	89.74	89.53	89.63	7.330×10^{-2}
$\text{BERT}_{\text{SDV}}(K=5)$	P. Corr	90.16	89.86	90.00	1.015×10^{-1}
	S. Corr	89.77	89.43	89.60	1.104×10^{-1}

Statistics of evaluating results have shown that our method has better performance and a smaller variance than the vanilla BERT fine-tuning. Since our self-ensemble mechanism inherits the property of the ensemble model, it is less sensitive to the data order than vanilla BERT fine-tuning.

4.5.2 *Convergence Curves*

In this subsection, we conduct experiments to analyze the effects of our models. The converge curve of the training phase is shown in Fig.4. According to the experimental results, fine-tuning $\text{BERT}_{\text{BASE}}$ cannot get significant improvement in the last two epochs (from 89.60 to 89.63). But with the help of the self-ensemble and the self-distillation mechanisms, the model keeps improving on the last two epochs. Similar observations can also be seen at IMDb.

To further analyze the reason for this observation, we also record the curve of the task-specific loss function $\mathcal{L}_{\text{ts}}(\cdot)$ and the self-distillation loss function $\mathcal{L}_{\text{sd}}(\cdot)$ on three datasets. As shown in Fig.5, the self-distillation loss increases at the beginning of the training phase. As the training processes, the self-distillation loss tends to oscillate. After that, the task-specific loss tends to converge on the last training steps, which cannot bring significant improvements. Therefore, adding the self-distillation loss function can bring more knowledge, continually improving the model even when the task-specific loss function cannot bring significant gains.

However, the shapes of the loss function curves are also related to the characteristics of the three datasets. For MRPC and RTE, the number of training examples is limited, which leads to a significant oscillation of self-distillation loss. Due to this observation, the weight of self-distillation loss λ should be related to the scale of the training set.

4.5.3 *Ablation Study*

In this subsection, we will compare our self-ensemble and self-distillation methods with other fine-tuning methods.

$\text{BERT}_{\text{BASE}}$. This method fine-tunes BERT-Base

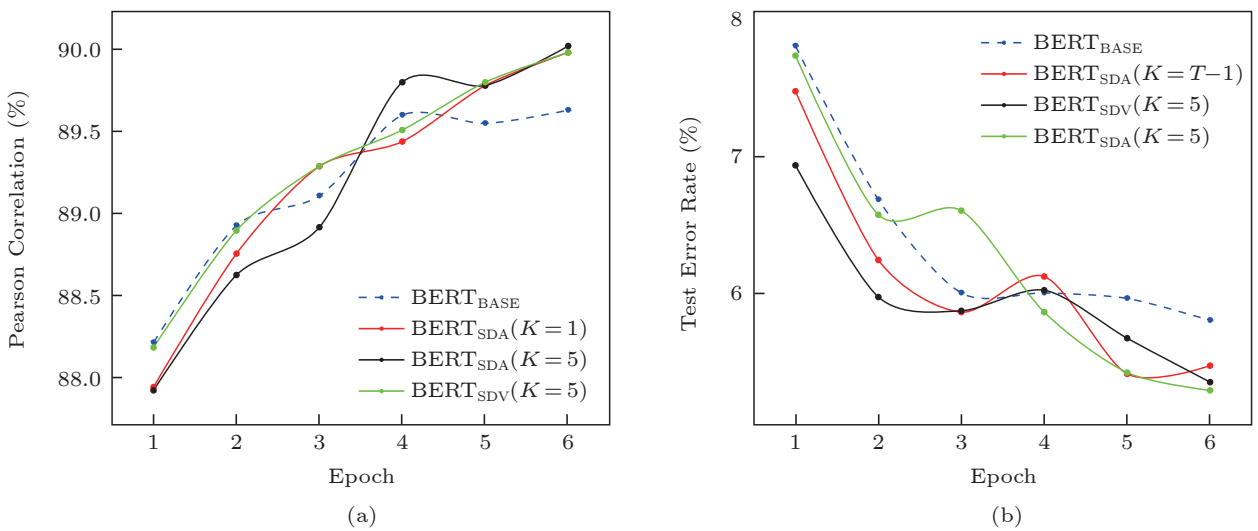


Fig.4. Convergence curves for different fine-tuning methods on (a) the development set of STS-B and (b) the test set of IMDb.

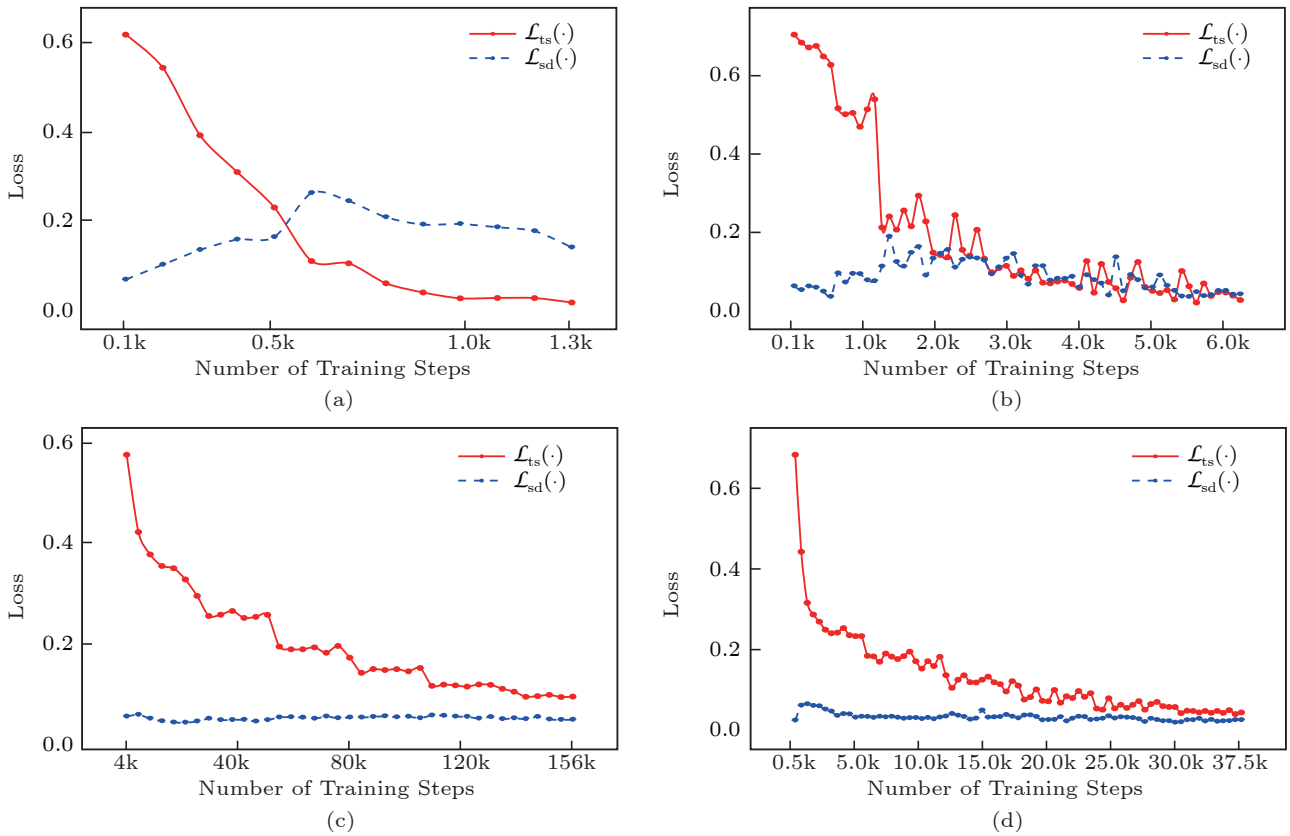


Fig.5. Loss curve of $\text{BERT}_{\text{SDA}}(K = 1)$ on four datasets. (a) MRPC. (b) RTE. (c) QNLI. (d) IMDb.

without any extra fine-tuning strategy or external data. It is our baseline model.

$BERT_{\text{VOTE}}$. This method fine-tunes K different BERT models with different initial random seeds and then sums up their probability distribution.

$BERT_{\text{AVG}}$. This method fine-tunes K different BERT models with different initial random seeds and then calculates the average of their parameters. The average parameters are used to conduct a new BERT model in the evaluation phase.

$BERT_{\text{SE}}$, $BERT_{\text{SDA}}$, and $BERT_{\text{SDV}}$. In this paper, we originally propose these methods. BERT_{SE} indicates only using self-ensemble (see Subsection 3.2). BERT_{SDA} and BERT_{SDV} indicate self-distillation-average (Subsection 3.3.1) and self-distillation-voted (Subsection 3.3.2), respectively.

As shown in Table 9, "$\text{BERT}_{\text{BASE}}$" indicates our baseline model, "$\text{BERT}_{\text{VOTE}}$" and "$\text{BERT}_{\text{AVG}}$" indicate traditional ensembles, "BERT_{SE}" indicates the method with only self-ensemble. In neural network ensemble, one of the most widely-used ensemble methods is the conclusion generation method that includes the absolute majority vote and the relative majority vote. Therefore the comparison with "$\text{BERT}_{\text{VOTE}}$" can be regarded as that with ensemble models. Compared with our baseline, self-ensemble has a slightly improvement in classification tasks. However, the method with only self-ensemble is worse than the traditional ensemble methods, which is the reason why we need self-distillation to further explore the potential of self-ensemble. In summary, on text classification tasks, SDA and SDV have a higher improvement on average (5.65% and 6.26%) versus traditional ensembles (5.44% and 4.07%).

4.5.4 *Discussion*

In this subsection, we will provide some discussion about our proposed self-ensemble and self-distillation methods.

Difference Between $BERT_{SDA}$ and $BERT_{SDV}$. We evaluate our proposed methods (SDA and SDV) on classification benchmarks and the GLUE benchmark. The results are shown in Table 9 and Table 6, respectively. Generally, NLI-like tasks (e.g., SNLI, QNLI, QQP, MNLI, and RTE) are pairwise-sentences classification tasks, in which BERT_{SDV} usually obtains better performance. On the other hand, in single-sentence classification tasks (e.g., IMDb, AG's News, DBPedia, Yelp), BERT_{SDA} usually obtains better per-

Table 9. Comparison of Fine-Tuning the BERT-Base ($\text{BERT}_{\text{BASE}}$) Model

Model	Test Error Rate (%)						Accuracy (%)	
	IMDb	AG's News	DBPedia	Yelp Polarity	Yelp Full	Avg. Δ	SNLI	Δ
ULMFiT[42]	4.60	5.01	0.80	2.16	29.98	/	/	/
$\text{BERT}_{\text{BASE}}$[13]*	5.40	5.25	0.71	2.28	30.06	/	/	/
$\text{BERT}_{\text{BASE}}$	5.80	5.71	0.71	2.25	30.37	-	90.7	-
$\text{BERT}_{\text{VOTE}}(K=4)$	5.60	5.41	0.67	2.03	29.44	5.44	91.2	5.50
$\text{BERT}_{\text{AVG}}(K=4)$	5.68	5.53	0.68	2.03	30.03	4.07	90.8	1.07
BERT_{SE}(ours)	5.82	5.59	0.65	2.19	30.48	2.50	90.8	1.07
BERT_{SDV}(ours)	5.35	5.38	**0.68**	2.05	**29.88**	5.65	**91.2**	**5.38**
BERT_{SDA}(ours)	**5.29**	**5.29**	**0.68**	**2.04**	**29.88**	**6.26**	91.2	5.38

Note: "*" indicates using extra fine-tuning strategies and data preprocessing. "/" means no available reported result. "Avg. Δ" means the average of relative changes.

formance. Based on this observation, we conclude that the pairwise-sentences classification tasks require more attention on segment-level interactions, which may be declined when applying parameter average.

Why Not the Latest Checkpoints? In order to prove the necessity of the experience pool, we conduct experiments on the AG's News dataset by using the latest checkpoints. In this experiment, the experience pool will remove the oldest checkpoint when a new checkpoint is added. The experimental result shows that the latest checkpoints perform worse: it gets the test error rate of 5.60% when the baseline of $\text{BERT}_{\text{BASE}}$ is 5.71% and the result of BERT_{SDA} is 5.29%. We observe that the difference of the latest checkpoints is limited because the optimization process would only change the parameter matrices slightly. Therefore, the latest checkpoints can be summarized in similar spaces, which reduce the diversity of the experience pool.

Comparison with Distillation-Based Methods. In order to compare the self-distillation with other distillation-based models, we conduct experiments in some GLUE datasets. In this experiment, we compare our model with MT-DNN[11] and MT-DNN$_{\text{KD}}$[32]. Results are shown in Table 10. In some small datasets such as CoLA, multi-task learning as well as knowledge distillation obtain a higher gain; therefore MT-DNN$_{\text{KD}}$ has an obvious improvement. In some large datasets such as SST-2, models learn much knowledge from the training set; therefore MT-DNN$_{\text{KD}}$ cannot provide an obvious improvement. Different from MT-DNN$_{\text{KD}}$, our proposed fine-tuning approach can provide improvements not only on small datasets but also on large datasets. In summary, our proposed fine-tuning approach gives a comparable performance of MT-DNN and MT-DNN$_{\text{KD}}$ on large datasets such as SST-2 and QNLI, while only obtains a similar performance with MT-DNN on small datasets such as CoLA. Due to the lack of external knowledge and the training data, the performances of BERT_{SDA} and BERT_{SDV} are acceptable compared with distillation-based models.

5 Conclusions

In this paper, we proposed a simple but effective fine-tuning approach for BERT without external knowledge or data. Specifically, we introduced two mechanisms: self-ensemble and self-distillation. The self-ensemble mechanism introduces the experience replay and establishes the teacher model with parameter averaging or logits voting. With self-distillation which leads to better teacher models, the students benefit from previous experience and become more robust. Experiments on the GLUE benchmark showed that our fine-tuning approach not only has a 0.9-point

Table 10. Comparison with Distillation-Based Methods on the Development Set of the GLUE Benchmark

Model	CoLA (Mcc)	SST-2 (Acc)	QQP (Acc/F_1)	MNLI-m/mm (Acc)	QNLI (Acc)
$\text{BERT}_{\text{LARGE}}$	61.8	93.5	91.1/88.0	86.3/86.2	92.4
MT-DNN[11]	63.5	94.3	91.9/89.2	87.1/86.7	92.9
MT-DNN$_{\text{KD}}$[32]	64.5	94.3	91.9/89.4	87.3/87.3	93.2
BERT_{SDA}(ours)	63.4	94.4	91.8/88.9	87.0/86.6	92.6
BERT_{SDV}(ours)	63.1	94.3	92.0/89.1	87.2/86.8	92.8

Note: Our proposed methods are initialized by 24-layer $\text{BERT}_{\text{LARGE}}$. Results of $\text{BERT}_{\text{LARGE}}$, MT-DNN, and MT-DNN$_{\text{KD}}$ are reported in [32].

performance boost compared with vanilla BERT, but also gets the gains on classification benchmarks. Our proposed approach also achieved a new state-of-the-art result on the SNLI dataset with the accuracy of 92.6%. Meanwhile, our proposed approach is orthogonal to the approaches with external data and knowledge. Therefore, we believe that more sophisticated hyperparameters and data augmentation can further boost our approach.

Conflict of Interest The authors declare that they have no conflict of interest.

References

[1] Devlin J, Chang M W, Lee K *et al.* BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies*, Jun. 2019, pp.4171–4186. DOI: 10.18653/v1/N19-1423.

[2] Yang Z L, Dai Z H, Yang Y M *et al.* XLNet: Generalized autoregressive pretraining for language understanding. In *Proc. the 33rd International Conference on Neural Information Processing Systems (NIPS)*, Dec. 2019, Article No. 517.

[3] Liu Y H, Ott M, Goyal N *et al.* RoBERTa: A robustly optimized BERT pretraining approach. arXiv: 1907.11692, 2019. https://arxiv.org/abs/1907.11692, Aug. 2023.

[4] Rajpurkar P, Zhang J, Lopyrev K *et al.* SQuAD: 100, 000+ questions for machine comprehension of text. In *Proc. the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Nov. 2016, pp.2383–2392. DOI: 10.18653/v1/D16-1264.

[5] Bowman S R, Angeli G, Potts C *et al.* A large annotated corpus for learning natural language inference. In *Proc. the 2015 EMNLP*, Sept. 2015, pp.632–642. DOI: 10.18653/v1/D15-1075.

[6] Qiu X P, Sun T X, Xu Y G *et al.* Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 2020, 63(10): 1872–1897. DOI: 10.1007/s11431-020-1647-3.

[7] Peters M E, Ruder S, Smith N A. To tune or not to tune? Adapting pretrained representations to diverse tasks. In *Proc. the 4th Workshop on Representation Learning for NLP*, Aug. 2019, pp.7–14. DOI: 10.18653/v1/W19-4302.

[8] Stickland A C, Murray I. BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. In *Proc. the 36th International Conference on Machine Learning (ICML)*, Jun. 2019, pp.5986–5995.

[9] Houlsby N, Giurgiu A, Jastrzebski S *et al.* Parameter-efficient transfer learning for NLP. In *Proc. the 36th ICML*, Jun. 2019, pp.2790–2799.

[10] Dong L, Yang N, Wang W H *et al.* Unified language model pre-training for natural language understanding and generation. arXiv: 1905.03197, 2019. https://arxiv.org/abs/1905.03197, Aug. 2023.

[11] Liu X D, He P C, Chen W Z *et al.* Multi-task deep neural networks for natural language understanding. arXiv: 1901.11504, 2019. https://arxiv.org/pdf/1901.11504.pdf, Aug. 2023.

[12] Raffel C, Shazeer N, Roberts A *et al.* Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv: 1910.10683, 2019. https://arxiv.org/abs/1910.10683, Aug. 2023.

[13] Sun C, Qiu X P, Xu Y G *et al.* How to fine-tune BERT for text classification? In *Proc. the 18th China National Conference on Chinese Computational Linguistics*, Oct. 2019, pp.194–206. DOI: 10.1007/978-3-030-32381-3_16.

[14] Li H, Wang X S, Ding S F. Research and development of neural network ensembles: A survey. *Artificial Intelligence Review*, 2018, 49(4): 455–479. DOI: 10.1007/s10462-016-9535-1.

[15] Polyak B T, Juditsky A B. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 1992, 30(4): 838–855. DOI: 10.1137/0330046.

[16] Schaul T, Quan J, Antonoglou I *et al.* Prioritized experience replay. In *Proc. the 4th International Conference on Learning Representations (ICLR)*, May 2016.

[17] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. arXiv: 1503.02531, 2015. https://arxiv.org/abs/1503.02531, Aug. 2023.

[18] Laine S, Aila T. Temporal ensembling for semi-supervised learning. In *Proc. the 5th ICLR*, Apr. 2017.

[19] Tarvainen A, Valpola H. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Proc. the 31st NIPS*, Dec. 2017, pp.1195–1204.

[20] Wei H R, Huang S J, Wang R *et al.* Online distilling from checkpoints for neural machine translation. In *Proc. the 2019 NAACL: Human Language Technologies*, Jun. 2019, pp.1932–1941. DOI: 10.18653/v1/N19-1192.

[21] Liu W J, Zhou P, Wang Z R *et al.* FastBERT: A self-distilling BERT with adaptive inference time. In *Proc. the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, Jul. 2020, pp.6035–6044. DOI: 10.18653/v1/2020.acl-main.537.

[22] Wang A, Singh A, Michael J *et al.* GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proc. the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Nov. 2018, pp.353–355. DOI: 10.18653/v1/W18-5446.

[23] Vaswani A, Shazeer N, Parmar N *et al.* Attention is all you need. In *Proc. the 31st NIPS*, Dec. 2017, pp.5998–6008.

[24] Sanh V, Debut L, Chaumond J *et al.* DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. arXiv: 1910.01108, 2019. https://arxiv.org/abs/1910.01108, Aug. 2023.

[25] Jiao X Q, Yin Y C, Shang L F *et al.* TinyBERT: Distilling BERT for natural language understanding. In *Proc. the 2020 Findings of the Association for Computational Linguistics*, Nov. 2020, pp.4163–4174. DOI: 10.18653/v1/2020.findings-emnlp.372.

[26] Sun Z Q, Yu H K, Song X D *et al.* MobileBERT: A compact task-agnostic BERT for resource-limited devices. In *Proc. the 58th ACL*, Jul. 2020, pp.2158–2170. DOI: 10.18653/v1/2020.acl-main.195.

[27] Wang W H, Wei F R, Dong L *et al.* MINILM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proc. the 34th NIPS*, Dec. 2020, Article No. 485.

[28] Ganaie M A, Hu M H, Malik A K *et al.* Ensemble deep learning: A review. *Eng. Appl. Artif. Intell.*, 2022, 105: 105–151. DOI: 10.1016/j.engappai.2022.105151.

[29] Andrychowicz M, Wolski F, Ray A *et al.* Hindsight experience replay. In *Proc. the 31st NIPS*, Dec. 2017, pp.5055–5065.

[30] Horgan D, Quan J, Budden D *et al.* Distributed prioritized experience replay. In *Proc. the 6th ICLR*, Apr. 30–May 3, 2018.

[31] Sun S Q, Cheng Y, Gan Z *et al.* Patient knowledge distillation for BERT model compression. In *Proc. the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, Nov. 2019, pp.4323–4332. DOI: 10.18653/v1/D19-1441.

[32] Liu X D, He P C, Chen W Z *et al.* Improving multi-task deep neural networks via knowledge distillation for natural language understanding. arXiv: 1904.09482, 2019. https://arxiv.org/abs/1904.09482, Aug. 2023.

[33] Warstadt A, Singh A, Bowman S R. Neural network acceptability judgments. *Trans. Association for Computational Linguistics*, 2019, 7: 625–641. DOI: 10.1162/tacl_a_00290.

[34] Socher R, Perelygin A, Wu J *et al.* Recursive deep models for semantic compositionality over a sentiment Treebank. In *Proc. EMNLP*, Oct. 2013, pp.1631–1642.

[35] Dolan W B, Brockett C. Automatically constructing a corpus of sentential paraphrases. In *Proc. the 3rd International Workshop on Paraphrasing*, Oct. 2005, pp.9–16.

[36] Cer D, Diab M, Agirre E *et al.* SemEval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. arXiv: 1708.00055, 2017. https://arxiv.org/abs/1708.00055, Aug. 2023.

[37] Williams A, Nangia N, Bowman S. A broad-coverage challenge corpus for sentence understanding through inference. In *Proc. the 2018 NAACL: Human Language Technologies*, Jun. 2018, pp.1112–1122. DOI: 10.18653/v1/N18-1101.

[38] Matthews B W. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 1975, 405(2): 442–451. DOI: 10.1016/0005-2795(75)90109-9.

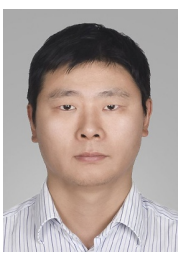
[39] Maas A L, Daly R E, Pham P T *et al.* Learning word vectors for sentiment analysis. In *Proc. the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Jun. 2011, pp.142–150.

[40] Zhang X, Zhao J B, LeCun Y. Character-level convolutional networks for text classification. In *Proc. the 28th NIPS*, Dec. 2015, pp.649–657.

[41] Pilault J, Elhattami A, Pal C. Conditionally adaptive multi-task learning: Improving transfer learning in NLP using fewer parameters & less data. arXiv: 2009.09139, 2020. https://arxiv.org/abs/2009.09139, Aug. 2023.

[42] Howard J, Ruder S. Universal language model fine-tuning for text classification. In *Proc. the 56th ACL*, Jul. 2018, pp.328–339. DOI: 10.18653/v1/P18-1031.

Yi-Ge Xu received his B.E. degree in computer science and technology from Shandong University, Jinan, in 2018. Currently he is a Master student in School of Computer Science, Fudan University, Shanghai. His research interests include natural language processing and deep learning.



Xi-Peng Qiu received his B.S. degree in chemistry and his Ph.D. degree in computer science from Fudan University, Shanghai, in 2001 and 2006 respectively. Currently he is a professor in School of Computer Science, Fudan University, Shanghai. His research interests include natural language processing and deep learning.

Li-Gao Zhou received his B.E. degree in electrical engineering from Huazhong University of Science and Technology, Wuhan, in 2007, and his M.E. degree in electrical engineering from Tsinghua University, Beijing, in 2010. Currently he is an expert in HUAWEI. His research interests include natural language processing and deep learning.

Xuan-Jing Huang received her B.S. and Ph.D. degrees in computer science from Fudan University, Shanghai, in 1993 and 1998 respectively. Currently she is a professor in School of Computer Science, Fudan University, Shanghai. Her research interests include natural language processing and deep learning.