applied sciences

MDPI

Article

MicroBERT: Distilling MoE-Based Knowledge from BERT into a Lighter Model

Dashun Zheng , Jiaxuan Li , Yunchu Yang, Yapeng Wang and Patrick Cheong-Iao Pang *

Faculty of Applied Sciences, Macao Polytechnic University, Macao 999078, China; p2212871@mpu.edu.mo (D.Z.); p2109376@mpu.edu.mo (J.L.); p1507937@mpu.edu.mo (Y.Y.)
* Correspondence: mail@patrickpang.net

Abstract: Natural language-processing tasks have been improved greatly by large language models (LLMs). However, numerous parameters make their execution computationally expensive and difficult on resource-constrained devices. For this problem, as well as maintaining accuracy, some techniques such as distillation and quantization have been proposed. Unfortunately, current methods fail to integrate model pruning with downstream tasks and overlook sentence-level semantic modeling, resulting in reduced efficiency of distillation. To alleviate these limitations, we propose a novel distilled lightweight model for BERT named MicroBERT. This method can transfer the knowledge contained in the "teacher" BERT model to a "student" BERT model. The sentence-level feature alignment loss (FAL) distillation mechanism, guided by Mixture-of-Experts (MoE), captures comprehensive contextual semantic knowledge from the "teacher" model to enhance the "student" model's performance while reducing its parameters. To make the outputs of "teacher" and "student" models comparable, we introduce the idea of a generative adversarial network (GAN) to train a discriminator. Our experimental results based on four datasets show that all steps of our distillation mechanism are effective, and the MicroBERT (101.14%) model outperforms TinyBERT (99%) by 2.24% in terms of average distillation reductions in various tasks on the GLUE dataset.

Keywords: natural language processing; knowledge distillation; generative adversarial networks; Mixture-of-Experts

check for updates

Citation: Zheng, D.; Li, J.; Yang, Y.; Wang, Y.; Pang, P.C.-I. MicroBERT: Distilling MoE-Based Knowledge from BERT into a Lighter Model. *Appl. Sci.* **2024**, *14*, 6171. https://doi.org/10.3390/app14146171

Academic Editors: Alaa Mohasseb and Andreas Kanavos

Received: 15 June 2024
Revised: 10 July 2024
Accepted: 10 July 2024
Published: 16 July 2024

1. Introduction

Research on pre-trained models has shown a trend towards increasingly larger models. Due to the proliferation of parameters, models with a vast number of parameters are now referred to as Large Language Models (LLMs). BERT [1], first proposed by Google in 2018, demonstrated the power of LLMs by achieving pioneering results on 11 NLP tasks. However, LLMs require massive training data for their enormous number of parameters, as evidenced by models such as BERT-base (109 M), BERT-Large (340 M), GPT-2 (1.5 B), and GPT-3 (175 B). The computational requirements for training and deploying these models often exceed the capabilities of typical academic research and commercial computing resources. For instance, the GPT-3 model proposed by OpenAI reportedly required 30,000 A100 GPUs for training. At this scale, the hardware costs alone could exceed millions of dollars, not including the high maintenance and electricity costs. Against this background, model compression techniques have developed rapidly in recent years. These include hierarchical quantization [2], weight pruning [3], and knowledge distillation (KD) [4]. These techniques aim to shorten inference time and reduce model size while maintaining accuracy, offering great value for both academic research and commercial applications. Currently, many studies focus on compressing BERT models based on KD frameworks [5–10]. While these distillation results generally meet requirements, they primarily focus on word-based features in the knowledge-distillation framework. This approach often ignores the contextual semantic information of the input text [11–13], which can lead to suboptimal results for tasks that rely heavily on contextual semantics, such as text classification, especially when distilling smaller models.

BERT processes input sequences of word vectors and prepends a [CLS] token. This design supervises word features but increases computational burden. KD aims to transfer knowledge to a lighter student network. We propose MicroBERT, a BERT-based KD model, with distinct loss functions for different BERT layers. We enhance feature construction in the hidden layer, prioritizing sentence features and reducing computation without losing word information. The prediction layer loss function serves as a soft target for richer information applicable to any task.

Two crucial components that affect the efficiency and performance of deep learning model training are the model's scale and the volume of the training data. Recently, a new training technique has been proposed and made accessible to the industry to address this problem. Larger models based on the Mixture of Experts (MoE) architecture [14] are more useful and efficient because the sparsely activated model adaptively selects a subset of its parameters for training based on the input data, whereas the densely activated model uses all of its parameters in computation. Furthermore, it is possible to scale the model parameters linearly without requiring additional computations.

The loss function between the prediction layers of the teacher model and the student model can generate richer information and be adapted to various tasks within the prediction layer. By employing the router as a soft target to choose the expert layer, the efficacy of model training is enhanced, thereby refining the model.

Finally, we introduce the idea of GAN [15], where the student model is used as a generator and the output of the student model is used as the adversarial sample generated by the generator, and the discriminator is trained using the predicted output of the teacher model and the student model. The ultimate goal of training the discriminator is to make it difficult to distinguish the output of the teacher model and the student model, i.e., to make the output of the student model as close as possible to the output of the teacher model. In addition, we devise a new feature-mapping method that allows the teacher model to pass information from all layers to the student model, further ensuring the integrity of knowledge transfer.

To demonstrate the relevant performance of our MicroBERT and the effectiveness of the algorithm, we conducted a number of experiments, including the performance of MicroBERT and TinyBERT on the General Language Understanding Evaluation (GLUE) [16] datasets with several different tasks and the distillation effect, the ablation experiments of two innovative distillation methods, and the comparison experiments. TinyBERT is not only one of the most commonly used models in industry and research communities but also TinyBERT has one of the highest citation rates in the field of knowledge distillation. The final ablation and comparison experiments demonstrate that our distillation steps are effective. The source code of this work can be found at: https://github.com/mpu-patrick-lab/microbert (accessed on 12 July 2024).

The main contributions of this paper are as follows:

- In order to promote the proper transfer of the linguistic information stored in the teacher's BERT to MicroBERT, we suggest a Transformer distillation approach called MicroBERT, with the routing algorithms for MoE models with reasonable serving costs. We investigate alternative routing methods that are taught to exploit global task-level knowledge to route all tokens predicted to a given task collectively to the same collection of experts.
- We propose a new feature-refinement method called Feature Alignment Loss (FAL), which enables the model to perform feature learning with a higher perspective, effectively reducing the computational burden of the model and shortening the inference time while ensuring accuracy.
- We design a new feature-mapping method to ensure that the information from the teacher model can be passed to the student model as much as possible, which improves

the efficiency of information utilization. At the same time, the soft target predicted for the teacher model can be migrated to any task.

- We introduced the idea of GAN to train discriminators using the outputs of the teacher model and the student model, so that the output of the student model can maximize the approximation of the output of the teacher model. This research aims to make the student model more lightweight by compressing the model without reducing the accuracy.

The remainder of this paper is organized as follows. We review existing model compression techniques and highlighting limitations in BERT-based KD frameworks. The Section 3 details MicroBERT's distillation process, including FAL, efficient inference using MoE, and GAN-based discriminator loss. Then, we present experiments comparing MicroBERT's accuracy with other models and analyzing the contribution of each component. The discussion highlights MicroBERT's effectiveness in accuracy and efficiency. Finally, we summarize our findings and suggest future research directions.

2. Related Work

In recent years, large-scale pre-trained language models such as BERT, GPT [17], ALBERT [18], and RoBERTa [19] and their derivatives have made significant progress in the field of NLP. However, these models usually require large-scale computational resources and extremely long inference times, leading to expensive computational costs. To improve inference speed and reduce redundant computations, a series of model compression methods such as distillation, quantization, and pruning have been proposed. For example, BERT-PKD [20] performs incremental knowledge extraction by extracting the output of the last layer of the model and learning from multiple intermediate layers, which can leverage rich information in the hidden layers of the teacher model and encourage the student model to patiently think about teacher learning and imitation through a multi-layer distillation process. DistilBERT [21] solves the problem by using knowledge distillation in the pre-training phase using knowledge distillation to solve the problem of previous work that only distilled for specific tasks. This ultimately reduces the parameters of the BERT model to 40% and achieves a 60% improvement in computational speed. DocBERT [22] is the first to apply BERT to the document classification task and distill BERT into a small bidirectional long short-term memory network LSTM using knowledge-distillation technology. MobileBERT [23] is model-independent, and by using a bottleneck structure based on BERT-large and striking a better balance between self-attentiveness and feedforward networks, the model becomes less parameterized at each layer which allows the depth of the model and BERT is consistent. In contrast, TinyBERT [24] introduces a frame to learn with two stages, which can simultaneously distill model pre-training and fine-tuning phases, and capture both generic semantic and task-specific knowledge, so that a new Transformer distillation method has appeared. ColBERT [25] introduced a post-interaction architecture that uses BERT to encode query and document independently and models their fine-grained similarity using a powerful interaction step that can effectively exploit the power of deep LM. MiniLM [26] used the student model to mimic the teacher model's self-attention module to train small models, which can effectively reduce the model parameters. FastBERT [27], proposed by Liu et al., divided the early existing classifiers into student and teacher classifiers and trained all student classifiers using the self-distillation technique, which also achieved good results.

Throughout all the model distillation frameworks nowadays, they only focus is on the extraction of word-level features while ignoring sentence-level features. At the same time, some models require additional data for finetuning to guarantee the model's capability after distillation, and other distillation methods fail to make a significant enough decrease in the models after distillation. The MoEBERT model [28] uses a hybrid expert structure and layer-by-layer distillation method to improve the ability and reasoning speed of BERT. Task-MoE [29] comprises substitute routing techniques that are taught to utilize global task-level data in order to route all tokens associated with a certain job to the same group of experts together. We decode distinct tasks independently, loading just the subset of task-specific experts during inference.

Therefore, we improve the performance of the model by introducing a sentence-level distillation mechanism to comprehensively learn generic contextual semantic knowledge from the teacher model, while reducing the parameters of the student model. Unlike traditional word-level distillation approaches, our method enhances the capture of contextual semantics, crucial for tasks heavily reliant on sentence understanding. This approach not only maintains high accuracy but also significantly reduces inference time and computational complexity.

3. Methods

In this section, we propose a generic knowledge-extraction method based on a Transformer model and design a discriminator for this model, through which we hope to make the results of the student model more accurate, which is called MicroBERT. The architecture diagram is shown in Figure 1.

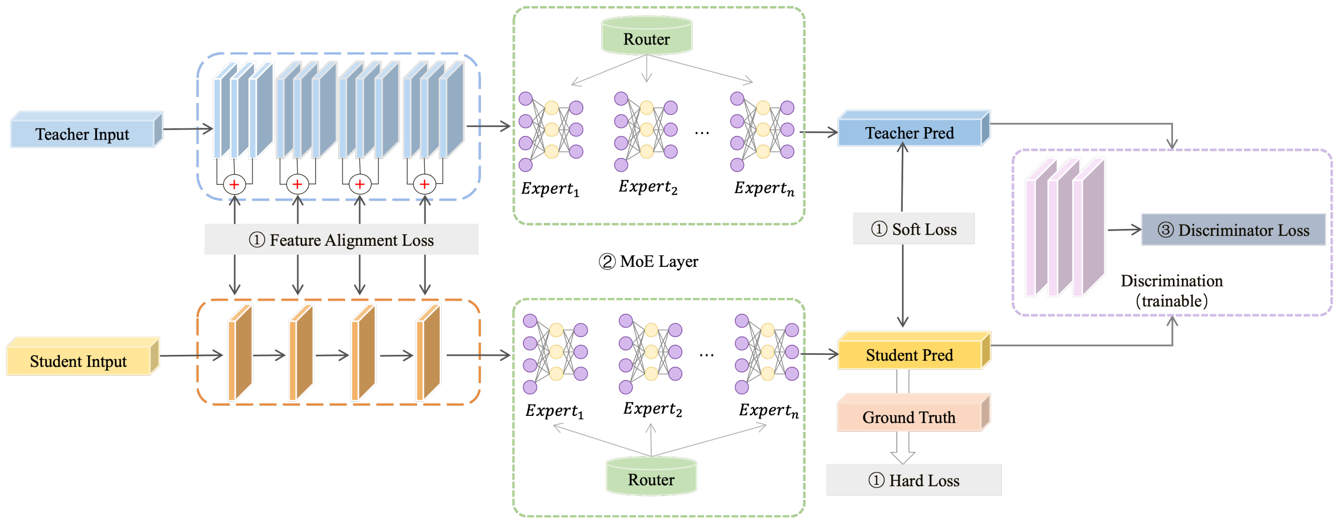

Figure 1. Overall Architecture Diagram: This diagram outlines the sophisticated design of the MicroBERT framework. ① The student model aligns its intermediate features with those of the teacher model using FAL. The predict layer is optimized through the application of soft loss derived from the teacher's outputs and hard loss based on the ground truth labels. ② Both models incorporate Mixture of Experts (MoE) layers, guided by routers, to enhance processing capabilities. ③ Furthermore, a discriminator network introduces discriminator loss to ensure the student's outputs closely resemble those of the teacher.

3.1. Distillation Process

We introduce a comprehensive framework for distillation training that can be applied to various transformer-based teacher and student models. Unlike traditional knowledge-distillation approaches, our framework enables supervision over both intermediate feature layers and the final output. In contrast, conventional approaches primarily focus on word-level features, resulting in redundancy and insufficient attention to sentence vectors. To address these limitations, we propose a distillation process that emphasizes both word vector features and sentence vector features, thereby enhancing overall efficiency.

In our method, we assume that the number of layers in the teacher model is N ($N > M$), while we set the number of layers in the student model to M. We want the student model to learn the content of the middle layer of the teacher model. Therefore, we extract the M layer from the teacher model to distill the knowledge to the m layer of the student model. The mapping formula is $n = g(m)$, which represents matching the m layer of the student model to the $g(m)$ layer of the teacher model. We specifically specify layer $M + 1$ as the prediction layer and layer 0 as the word embedding layer.

We derived two formulas, $g(0) = 0$ and $g(M+1) = N+1$, where $g(m)$ represents the mapping layer between the teacher model and the student model. To determine the optimal number of mapping layers, we conducted a series of comparative experiments to evaluate the impact of different choices for $g(m)$ on the overall model performance. This layer-mapping knowledge-distillation method effectively reduces the disparity between student and teacher models, enabling selective acquisition of relevant knowledge from the teacher model. As a result, knowledge distillation is performed more efficiently. Our framework provides a layer-mapping approach that significantly enhances the effectiveness of knowledge distillation. We therefore derive the following objectives:

$$\mathcal{L}_{\text{model}} = \sum_{x \in \mathcal{X}} \sum_{m=0}^{M+1} \lambda_m \mathcal{L}_{\text{layer}}\left(f^T_{g(m)}(x), f^S_m(x)\right) \tag{1}$$

where $f_m(x)$ is the behavior function of m layer, λ_m is the hyperparameter, the main purpose of which is to change the weight of the distillation. $\mathcal{L}_{\text{layer}}$ is the loss function of the provided model layer.

Hidden Layer Distillation. Our proposed distillation technique for transformer layers builds upon hidden state distillation. Recent research has shown that the [CLS] token, max pooling and mean pooling, which BERT learns, possess robust sentence vector representations. Consequently, distilling such information becomes a crucial factor for sentence vector-based distillation. This linguistic knowledge encompasses syntactic information and other essential aspects for natural language understanding. We anticipate that the student model (MicroBERT) will learn more semantic knowledge from the teacher model (BERT). Therefore, we introduce a hidden layer-based language-refinement method called FAL. We extract three aspects from FAL: the [CLS] token, mean pooling and max pooling. These aspects are then trained by the students, enabling the student model to focus not only on word vector features but also on sentence vector features. The following objectives have been defined:

$$\mathcal{L}_{\text{hidn}} = \text{MSE}\left(\boldsymbol{H}^T, \boldsymbol{H}^S \boldsymbol{W}_h\right) \tag{2}$$

where $\boldsymbol{H}^S \in \mathbb{R}^{l \times d'}$ and $\boldsymbol{H}^T \in \mathbb{R}^{l \times d}$ are matrices that represent the hidden states of the student and teacher models, as defined by Equation (2). Thus the hidden sizes of the classroom model and student model are defined as d and d'. We want to maximize the compression of the model, so the student model d' is chosen to be frequently smaller than d. Also, the hidden state of the student model is transformed to the same state dimension as the teacher model by the learnable linear transformation matrix mapping $\boldsymbol{W}_h \in \mathbb{R}^{d' \times d}$.

Embedding Layer Distillation. We also performed distillation on the embedding layer, aiming to achieve similar goals as the hidden-state-based distillation:

$$\mathcal{L}_{\text{embd}} = \text{MSE}\left(\boldsymbol{E}^T, \boldsymbol{E}^S \boldsymbol{W}_e\right) \tag{3}$$

where the matrices $\boldsymbol{E}^S$ and $\boldsymbol{E}^T$ stand for the networks' respective embeddings in students' and teachers' networks. They share the same structure as the hidden state matrices in this study. An analogous linear transformation to $\boldsymbol{W}_h$ is the matrix $\boldsymbol{W}_e$.

MoE-guided distillation. We use the MoE proposed by LLM [30], where the MoE layers for the Transformers consist of E Prediction Layer, such that $(PL_1 \ldots PL_E)$.

$$PL_e(x_s) = wo_e Softmax(wi_e x_s) \tag{4}$$

$$y_s = \sum_{e=1}^{E} G_{s,e} PL_e(x_s) \tag{5}$$

The input token for the MoE layer at position s is x_s, and each PL_e denotes a one-layer neural network with a Softmax activation function. The projection weights for the e-th expert's input and output are wi_e and wo_e. Finally, $G_{s,e}$ is the vector produced by the gating network, commonly referred to as the router. For each expert, the vector's values are mostly zeros, with the exception of one positive value. Using this vector, we route the token to a restricted set of experts. The items chosen from G_s, E, determine the expert's contribution to the final output y_s. It should be noted that in order to be comparable with the previous study, we selected the top 1 weight experts for each case in this work.

The gating network G_s, E, where (1) expert utilization must be balanced and (2) the function must be efficient to execute at scale, must be carefully examined for efficiency reasons [30].

Prediction Layer Distillation. For the prediction layer component of our model, we employ the knowledge-distillation method proposed by Hinton et al. [31]. This method involves calculating the soft cross-entropy loss by taking the logarithm of the outputs from both the student model and the teacher model. Additionally, the soft labels generated by the trained teacher model can be transferred to any task.

$$\mathcal{L}_{\text{pred}} = \text{CE}\left(z^T/t, z^S/t\right) \tag{6}$$

where CE stands for cross entropy loss, t stands for temperature value and z^S and z^T are the logits vectors predicted by the student and teacher, respectively. In our experiment, t = 1 shows good performance.

Finally, the above Equations (2), (3) and (6) distillation losses are summarized to yield the following target losses:

$$\mathcal{L}_{\text{layer}} = \mathcal{L}_{\text{embd}} + \mathcal{L}_{\text{hidn}} + \mathcal{L}_{\text{pred}} \tag{7}$$

where calculate $\mathcal{L}_{\text{embd}}$ loss if $m = 0$, $\mathcal{L}_{\text{hidn}}$ loss if $M \geq m > 0$ and $\mathcal{L}_{\text{pred}}$ loss if $m = M + 1$.

3.2. Feature Alignment Loss

In this section, we introduce a loss function called FAL. Our objective in designing this function serves two purposes. Firstly, we aim for the smaller model to learn the comprehensive knowledge of the middle layer from the larger model. Secondly, we intend for the model to prioritize the word vectors and sentence vectors, regardless of the word vectors. The loss function we designed is shown in Figure 2.

In this paper, we investigate the application of the BERT model for feature extraction. In the BERT$_{\text{base}}$ model, the BERT model takes into account that many downstream tasks rely on analyzing the relationship between two sentences for modeling, such as question answering. To enable the model to have this capability, for such tasks, BERT does so by splicing a [CLS] token at the beginning of a sentence. [CLS] token is encoded by BERT and the resulting vector representation is usually used as the representation of the current sentence. In addition, the BERT model takes word sequences as input and passes them up through multiple layers of encoders. Each layer passes through Self-Attention and Feedforward Neural Network (FFNN). The encoding of all tokens by BERT outputs a vector of size hidden size (768 in BERT$_{\text{base}}$) at each position.

Since the vector representation of all tokens is available to BERT, it is possible to use the vectors of these tokens directly as features and feed them into a task-specific neural network for training. In addition, the output of the last layer of BERT can also be used to connect the task network for fine-tuning. However, traditional BERT has some problems, such as the need to supervise the features of each word while ignoring the sentence vector features. This not only results in generating excessive computational effort but also reduces the generalization of the model.

At the same time, the implied dimension of the BERT$_{\text{base}}$ is 768, while the implied dimension of our MicroBERT student model is 312. Due to the mismatch of these two dimensions, the distillation operation cannot be performed. To solve the above problem, we map the two features. Then the [CLS] token in the features is taken out, the mean pooling and the max pooling are taken out and the loss between them is calculated, which is the FAL.

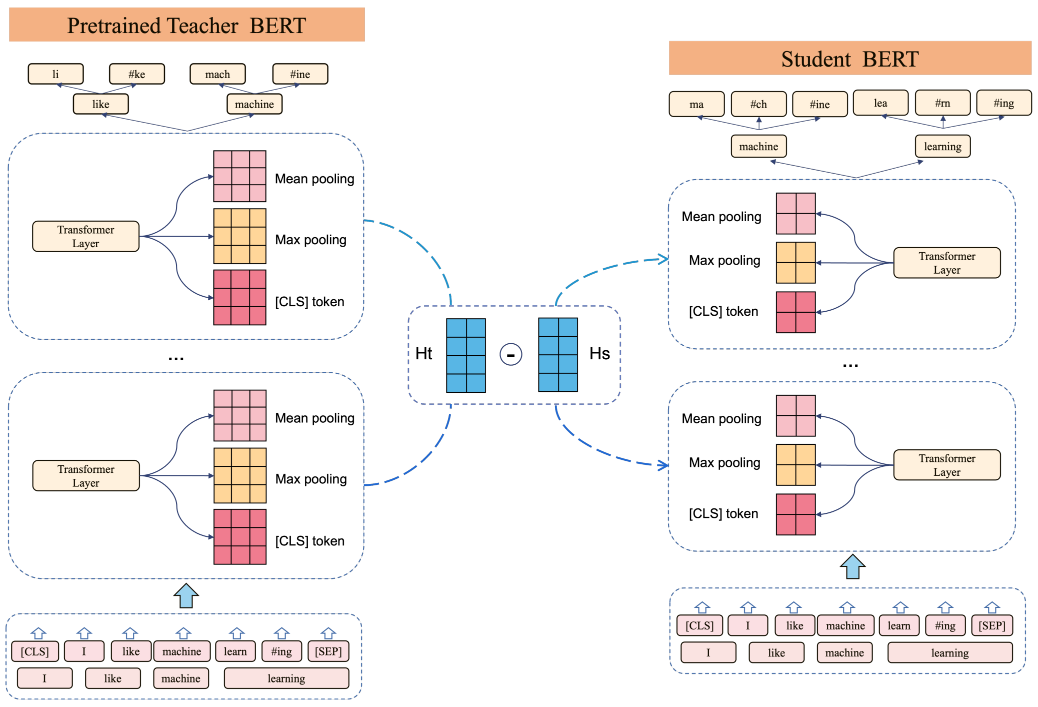

Figure 2. FAL: This diagram illustrates the feature alignment process between a pretrained Teacher BERT and a Student BERT. Both models process input tokens through their respective transformer layers. The outputs are then aggregated using mean pooling, max pooling and the [CLS] token. These features are aligned by minimizing the difference between corresponding outputs from the teacher and student models.

We concatenate the three parts, the [CLS] token features, the mean pooled features and the maximum pooled features, and use these three parts as features for distillation training inside supervised learning, with the mean and maximum equivalent to the pooled elemental features. The advantage of this is that we aim to make the model not only reduce the computational work but also focus on the features of the sentence vectors and word vectors.

For each layer of BERT, we obtain the features of [CLS] token, the features of mean pooling and the features of max pooling, and concatenate them together:

$$\mathcal{L}_{FAL} = [CLS]token + Mean\ pooling + Max\ pooling \tag{8}$$

The purpose of this is to ensure that the number of layers of BERT$_{\text{base}}$ corresponds to the number of layers of MicroBERT to find the FAL.

For the [CLS] token, we used two feature-extraction methods, the first one is the MSE loss calculation for layers 3, 6, 9 and 12 of the teacher model corresponding to layers 1, 2, 3 and 4 of the student model, and the second one is the averaging method, where the MSE loss calculation is performed by averaging layers 1–3, 3–6, 6–9 and 9–12 of the BERT with layers 1, 2, 3 and 4 of the MicroBERT. Meanwhile, the experimental results prove that the second method is better, and we consider the second method includes the information transfer of all layers and conveys more information, so it will have better results.

3.3. *Soft Loss and Hard Loss*

Soft Loss (SL) and Hard Loss (HL) are present in the traditional knowledge-distillation method. Because the traditional method of neural network training is to define a loss function so that the loss function is as small as possible, in order to predict values as close to the true values as possible, this training process is to find the maximum likelihood estimate for the ground truth. In knowledge distillation, the teacher model's softmax layer output is used as a soft-target to augment the student model's hard-target training. This has the advantage of allowing the data to carry substantial information from the teacher model's predictions, beyond just the positive labels. This way the student model gets more information than the traditional training method.

The weight of HL is fixed to 1, and the weight coefficients of FAL, SL and HL are set as hyperparameters for adjustment, and the final weighted sum of FAL, SL and HL is weighted, multi-loss joint training total-loss = FAL + SL + HL and DL is used as a The discriminator is used to distinguish whether the model is from a (BERT$_{\text{base}}$) teacher model or a student model and the step is to train the discriminator first and then to train the GAN.

$$\mathcal{L}_{total} = \alpha\mathcal{L}_{soft} + \beta\mathcal{L}_{hard} + \gamma\mathcal{L}_{FAL} \tag{9}$$

where $\mathcal{L}_{soft}$ denotes the dispersion between the probability distributions of the teacher and student network outputs, $\mathcal{L}_{hard}$ denotes the cross entropy of the label of the input image and the probability distribution of the network output, that is, the loss between the probability distribution of the softmax output and the label. And γ is another hyperparameter to measure the importance of the distillation characteristics of the middle layer.

3.4. Efficient Inference on MoE

The MoE training component uses dynamic graph training, which is resilient and adaptable [14]. On the other hand, the inference process uses the neural network layer to achieve stability and efficiency. There are three phases in the inference process altogether (shown in Figure 3):

- Fusion: The origin graph and the matching distributed method for the ultra-large-scale distributed training model are combined to eliminate parameter redundancy.
- Distillation and Compression: Less experts are present in the student network as a result of the instructor network's many experts being concentrated and condensed.
- Optimization: Relevant IR Pass optimizations, including kernel fusion, are applied to the distributed sub-graphs in order to further increase the inference time.

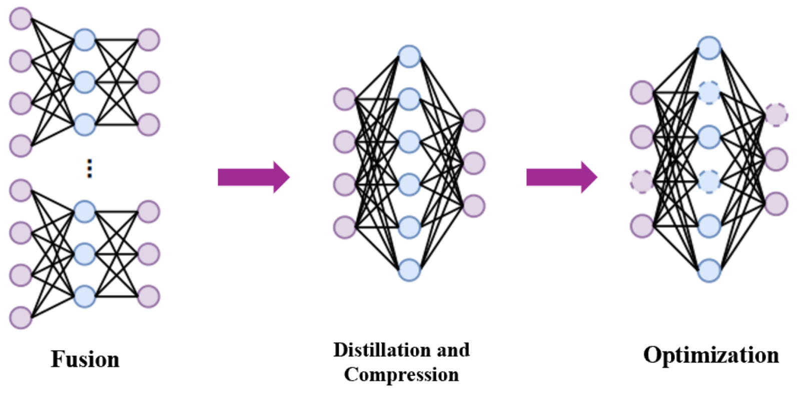

Figure 3. Efficient Inference on MoE: This diagram illustrates the process within the Mixture of Experts (MoE) layer. It begins with the fusion of multiple expert networks, followed by distillation and compression to consolidate their outputs. The final step involves optimization, where the combined network is fine-tuned for improved performance.

3.5. Discriminator Loss

A trainable discriminator is utilized to compute the Discriminator Loss (DL). In this context, we introduce the concept of a Generative Adversarial Network (GAN), where the generator takes the role of the student model. The primary objective of the generator is to create adversarial samples. The discriminator is trained using the anticipated output of the instructor model, which serves as its training input, ultimately leading to the acquisition of DL. Ultimately, we obtain a trained discriminator capable of distinguishing between the teacher and student models based on the origin of the projected data. Discrimination is crucial to ensure that the student model learns meaningful representations from the input, rather than merely memorizing the teacher's output.

As seen in Equation (10) below, the process of finding a binary function's minimum may be characterized as GAN training.

$$\min_S \max_D \mathcal{L}(D,S) = \min_S \max_D \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(T(x))] + \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log(1 - D(S(x)))] \tag{10}$$

Two goals are achieved by optimizing the loss function $\mathcal{L}(D,S)$: first, it motivates the generator S to provide genuine samples; second, it improves the discriminator D's capacity to distinguish between samples from Tea_Pred and Stu_Pred.

The generator is associated with the student model S, which attempts to replicate the teacher model T's output. Consequently, Equation (11) defines the loss function of the student model S.

$$\mathcal{L}_S = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log(1 - D(S(x)))] \tag{11}$$

While the student model's output aims to mimic the instructor model in order to trick the discriminator, the teacher model's output is recognized as authentic data. Thus, Equation (12) may be used to define the discriminator D's loss function.

$$\mathcal{L}_D = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(T(x))] + \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log(1 - D(S(x)))] \tag{12}$$

The average of the binary cross-entropy (BCE) losses corresponding to the teacher and student models' outputs is the discriminator D's loss function. The discriminator D will be trained using this loss function in order to improve the ability to discriminate between the outputs of the student and instructor models. As indicated in Equation (13).

$$\begin{aligned}
&\max_D \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(T(x))] + \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log(1 - D(S(x)))] \\
\Rightarrow &\min_D \mathbb{E}_{x\sim p_{\text{data}}(x)}[-\log D(T(x))] + \mathbb{E}_{x\sim p_{\text{data}}(x)}[-\log(1 - D(S(x)))] \\
\Rightarrow &\min_D \mathbb{E}_{x\sim p_{\text{data}}(x)}[\text{BCE}(D(T(x)),1)] + \mathbb{E}_{x\sim p_{\text{data}}(x)}[\text{BCE}(D(S(x)),0)] \\
\Rightarrow &\min_D \frac{1}{N}\sum_{i=1}^{N}[\text{BCE}(D(T(x_i)),1) + \text{BCE}(D(S(x_i)),0)]
\end{aligned} \tag{13}$$

where $T(x)$ represents the output of the teacher model on real data sample x and $S(x)$ represents the corresponding output generated by the student model, the discriminator's loss function aims to distinguish between these two sources of information. As derived above, the maximization objective for the discriminator D in a knowledge-distillation setting can be reformulated as minimizing the sum of two BCE terms. Specifically, the first BCE term quantifies the discriminator's ability to recognize the teacher model's outputs as authentic (i.e., labeled as 1), while the second BCE term measures its performance in identifying the student model's outputs as generated (i.e., labeled as 0). By minimizing this combined loss, the discriminator learns to effectively distinguish between the outputs of the teacher and student models, thereby guiding the student towards better learning of the teacher's knowledge.

4. Experiments and Results

4.1. Model Accuracy Comparison Experiment

We construct a small student model MicroBERT$_4$ with 14.5 M parameters in total. It contains 12 attention heads (h = 12), 4 encoder layers (M = 4), a feed-forward/filter size of 1200 (d_i = 1200) and hidden dimensions of 312 (d = 312). The teacher model, which has 110 M parameters, is based on the original BERT$_{\text{base}}$ (d = 768, d_i = 3072, N = 12 and h = 12). We extracted the average taken every third layer as the layer-mapping function in the teacher model. so MicroBERT4 learns by taking the average from every 3 layers of BERT$_{\text{base}}$.

For training, PyTorch 1.7.0 was used to implement the model. This model was trained on a server configuration of NVIDIA GeForce RTX 3090 Ti purchased from Macao SAR, China. The algorithm was trained on Ubuntu 22.04, a 64-bit operating system. We performed knowledge distillation at the intermediate layer for {5, 10, 20} epochs using a batch size of 32. The batch size was selected from {16, 32} and the learning rate was selected from $\{1 \times 10^{-5}, 2 \times 10^{-5}, 3 \times 10^{-5}\}$. For the single-sentence task, the maximum sequence length during distillation was set to 64. For the sequence-pair task, the maximum sequence length was set to 128.

We propose a novel lightweight framework and new mapping methods. To successfully distill a more lightweight model than TinyBERT. To investigate whether our super lightweight model can maintain competitive accuracy, we first tested it on six different well-known NLP open source dataset tasks in GLUE, namely SST-2 for sentiment classification task, MNLI-M for semantic matching task, MRPC and QQP for semantic judgment task and RTE for binary classification task. In terms of models, in addition to our ultra-lightweight model, we also selected the initial large model, which is also the distilled teacher model BERT$_{\text{base}}$, and lightweight models for several different compression methods, and also for a fair comparison, we trained 4-layer BERT-PKD$_4$, 4-layer DistillBERT$_4$ and 4-layer MinilmV2$_4$ using the published code and fine-tuning these 4-layer baselines using the proposed hyper-parameters. The accuracy results for their dataset tasks are shown in the following Table 1.

Table 1. Model effect comparison table.

System	Params	FLOPs	Speedup	SST-2	MNLI-m	MRPC	QQP	RTE	Avg
BERT(Google)	109 M	22.5 B	1.0×	93.5	84.6	88.9	71.2	66.4	80.92
BERT(Teacher)	109 M	22.5 B	1.0×	91.1	**84.4**	86.0	**88.8**	64.2	**82.9**
BERT-PKD$_4$	52.2 M	7.6 B	3.0×	89.4	79.9	82.6	70.2	62.3	76.88
DistillBERT$_4$	52.2 M	7.6 B	3.0×	**91.4**	78.9	82.4	68.5	54.1	75.06
MobileBERT$_4$	15.1 M	3.1 B	-	91.2	81.5	87.9	68.9	**65.1**	78.92
TinyBERT$_4$	14.5 M	1.2 B	9.4×	87.6	80.5	82.3	87.7	61.7	79.96
MinilmV2$_4$	5.4 M	4.0 B	5.3×	88.4	74.3	81.8	82.0	60.6	78.0
MicroBERT	14.5 M	1.2 B	9.3×	89.6	80.3	**88.7**	86.6	62.8	81.6
Avg				89.8	79.9	84.5	78.9	61.9	79.2

On the GLUE benchmark test set, the models are assessed. For each model group, the top outcomes are bold. Using a single NVIDIA 3090 Ti GPU, inference speedup is evaluated.

In the five datasets selected for this study, accuracy (Acc) was used as the evaluation metric for the tasks SST-2, MNLI-m and RTE. Conversely, the F1-score was employed as the evaluation metric for the MRPC and QQP tasks.

Based on the results we found that in the SST-2 dataset for the sentiment classification task, the best result for DistillBERT reached 91.4%, which exceeded the accuracy of BERT (Teacher) by 0.3%. Within this task, our ultra-lightweight model was only 0.6% lower than the average accuracy of the five lightweight models, maintaining a strong competitive position. The best results in MNLI-m for the semantic type matching task were 81.5% for MobileBERT$_4$ and 84.4% for BERT (Teacher) and the accuracy of our ultra-lightweight model in this task is about 0.4% higher than the average accuracy of the four lightweight models, which is within our acceptance range. In MRPC for the semantic judgment task, the best models are 88.7% for our ultra-lightweight model, reaching an extremely high level. The 86.6% in QQP is about 6.5% more than the average accuracy of the other lightweight models. Finally, in the RTE task, our model is about 3.6% less accurate than the best BERT (Google), while it is higher than the average accuracy.

In summary, based on the multi-task multi-model comparison experiments, we find that our ultra-lightweight model still maintains an extremely impressive accuracy rate in each task, and does not sacrifice too much in accuracy while improving the lightweight, and still remains competitive among the accuracy rates of the four lightweight models.

4.2. Distillation Effect Comparison Experiment

To demonstrate the distillation effect more intuitively in the experiment, as shown in Table 2. We used five different task datasets from the GLUE dataset for the experiment and counted the accuracy of the student model and the teacher model separately. Secondly, the accuracy of the student model alone does not specifically represent the reduction of the distillation model, so we obtained the model reduction by dividing the accuracy of the student model by the accuracy of the teacher model, with a higher reduction representing better distillation. The results show that in the SST-2 dataset, our model still achieves 98.3% distillation reduction with a teacher model accuracy of 91.1%. In MNLI tasks, the model achieved a reduction rate of 95.1%, which is also the worst effect distillation reduction in all datasets, but the reduction rate of 95% for all tasks has achieved our expected performance. In the MRPC and QQP datasets, the model effects reached 99.7% and 97.5%, respectively. Finally, in the RTE dataset, the distillation of our model performed exceptionally well, with the accuracy of the student model surpassing the teacher's accuracy, reaching an amazing 114.5% reduction.

Table 2. Comparison table for reappear percent ability.

System	SST-2	MNLI-m	MRPC	QQP	RTE	Avg
MicroBERT$_{tea}$	91.1	84.4	88.9	88.8	54.8	-
MicroBERT$_{stu}$	89.6	80.3	88.7	86.6	62.8	-
Reappear persent	**98.3**	**95.1**	**99.7**	**97.5**	**114.5**	**101.0**
TinyBERT$_{tea}$	93.4	83.9	87.5	71.1	67.1	-
TinyBERT$_{stu}$	92.6	82.5	86.4	71.3	66.6	-
Reappear percent	**99.1**	**98.3**	**98.7**	**100.2**	**99.4**	**99.0**
Difference (Tiny-Micro)	**−0.79**	**−3.2**	**1.03**	**−2.76**	**15.1**	**−2.0**

Reappear percent is the distillation contrast between the teacher model and the student model with the formula stu/tea. Difference (Tiny-Micro) is the comparison between TinyBERT and MicroBERT, with the formula $tea - stu/tea$. The stu is student model accuracy and tea is teacher model accuracy.

In the experiments, just counting the reduction of our model could not objectively show the effect of model distillation, so we added the well-known distillation model TinyBERT, which outperformed our model by 2.76% in the QQP dataset in the same dataset, and also in the MNLI and SST-2 datasets. The restoration effect in MNLI and SST-2 datasets is also a little higher than our model, but both are less than 4%. Secondly, the reduction of TinyBERT in both MRPC and RTE datasets is lower than our model, especially in the RTE dataset; the reduction effect is more than 15% lower than our model and the difference is huge. Finally, we counted the average reduction of both models in the five dataset tasks, and the results showed that the average reduction of our model was 101.0%, and it exceeded the 99% reduction of TinyBERT, and this result proved that the distillation reduction of our model exceeded that of TinyBERT in the five datasets tasks.

4.3. Loss Algorithm Comparison Experiment

After proving the reliability of our model for multiple dataset tasks, we designed a comparison experiment of our algorithm to validate the effectiveness of one of the steps of our algorithm. As shown in Figure 4, in traditional distillation models, such as TinyBERT, they take one layer of loss at each of the four layers for the next step, as shown by taking layer 3, layer 6, layer 9 and layer 12. We believe that when we extract a layer, the first three layers of that layer will also have an impact on it, so we should take the average loss of its three layers for the operation. According to the experimental results, we can see that the accuracy of our average algorithm is higher than the previous traditional algorithm in all five different dataset tasks, and the accuracy is higher by more than 1% on average, which again proves that our idea is correct.

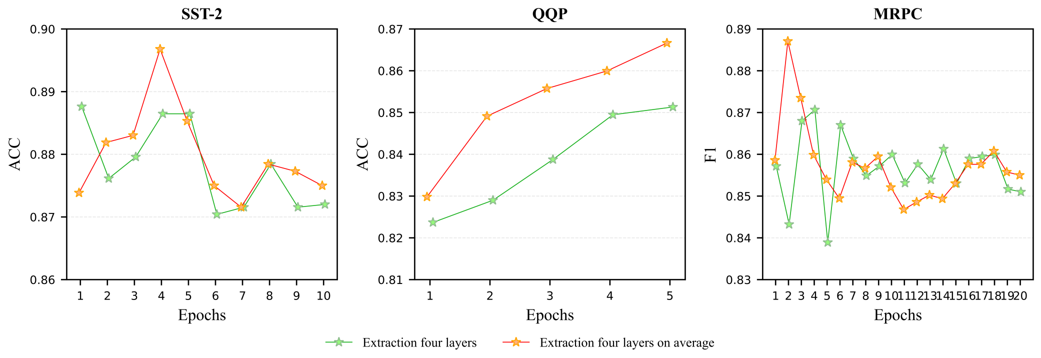

Figure 4. Compare FAL loss function selection methods using three datasets.

4.4. Algorithmic Ablation Experiments

We performed ablation experiments on our model to verify the validity of each step and method of our distillation model. As shown in Table 3. First, the algorithm accuracy decreased to different degrees in all three dataset tasks when we kept only the GAN operation, so we concluded that all operations except the GAN operation have validity because of the impact on the accuracy of the model. Secondly, we found that the accuracy of the algorithm decreases to different degrees in all five datasets, while keeping the rest of the operations and removing only the GAN operation, which proves the necessity of the GAN operation for the algorithm. Finally, we keep only the soft loss and the hard loss in the four loss tasks, and the results prove that there is also a decrease in all three datasets, which also verifies the validity of the two additional losses proposed by our model.

Table 3. The ablation experiment on MicroBERT's distillation on the Validation set.

System	SST-2	QQP	MRPC
our_model	89.6	86.6	88.7
w/o Dis	88.9	86.3	88.0
w/o Hidn	88.3	85.9	87.7
w/o Pred	87.4	85.1	87.1

Ablation studies of different distillation objectives in the MicroBERT learning. These variants were validated on the development set. w/o Dis removes the loss operation from the discriminator. w/o Hidn stands for removing the loss operation from the Hidden layer. w/o Pred is removing the loss operation from the Prediction layer.

5. Discussion

Our proposed MicroBERT model, which is lighter than existing models, incorporates a novel distillation algorithm, sentence-level FAL and a GAN-trained discriminator. Across various tasks, MicroBERT surpasses other lightweight models by 1.5% in accuracy while maintaining its lightweight status. In particular, it outperforms TinyBERT by 2% in distillation ability, showcasing its superiority in transferring knowledge from a large teacher model. Despite its lightweight nature, MicroBERT consistently maintains high accuracy, making it beneficial for various applications, especially in resource-constrained environments.

In current research, only elite institutions with substantial resources can sustain LLMs. However, MicroBERT addresses this issue by providing a small and efficient model that can be easily deployed locally, even in privacy-sensitive contexts. Its lightweight design allows for wider adoption and use in practical applications. Although MicroBERT significantly reduces parameters and computational costs, it is important to acknowledge the potential trade-offs in model robustness and generalizability. As with any model compression technique, there is a delicate balance between achieving efficiency and maintaining performance across diverse tasks and datasets. Further research is needed to fully understand these trade-offs and to develop strategies for mitigating potential robustness and generalizability issues in lightweight models like MicroBERT. Despite these potential challenges, our distillation approach offers a new perspective for advancing the field of LLMs and facilitating applications.

6. Conclusions and Future Work

In this research, we propose a novel Transformer-based distillation method that allows the model to learn features from both word and sentence vectors. Through numerous tests, we have observed that our method significantly reduces the model size and inference time without sacrificing high accuracy. This approach presents a practical solution for deploying $\text{BERT}_{\text{base}}$ NLP models on hardware. However, it is important to note that our study has certain limitations. Specifically, we have not explored the effectiveness of our method on models larger than $\text{BERT}_{\text{base}}$, such as BERT-large, and we have not conducted extensive experiments on languages other than English. Future research could delve into the methods of effectively transferring knowledge from a comprehensive and sophisticated teacher model, such as BERT-large, to a compact student model like MicroBERT, addressing these limitations. Additionally, an interesting avenue to further condense pre-trained language models is to combine distillation with quantization or pruning techniques. In conclusion, we highlight the importance of focusing future research on enhancing inference efficiency for large models by developing methods that are more inference-friendly while maintaining the quality improvements of MoE models. We believe that further studies on hierarchical variations or more detailed routing hybrids will yield additional benefits and deepen our understanding of large-scale, heavily multi-tasking and sparsely gated networks.

Author Contributions: D.Z.: Conceptualization, Methodology, Software, Writing—Original Draft, Visualization. J.L.: Methodology, Data Curation, Writing—Original Draft. Y.Y.: Validation, Investigation. Y.W.: Writing—Review & Editing. P.C.-I.P.: Supervision, Writing—Review & Editing. All authors have read and agreed to the published version of the manuscript.

Funding: The project by Macao Polytechnic University research grant (project code: RP/FCA-10/2022). Patrick Cheong-Iao Pang is partially supported by a funding from Macao Science and Technology Development Fund (funding ID: 0048/2021/APD).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset is publicly available at https://gluebenchmark.com/ (accessed on 20 November 2023).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
2. Gong, Y.; Liu, L.; Yang, M.; Bourdev, L. Compressing deep convolutional networks using vector quantization. *arXiv* **2014**, arXiv:1412.6115.
3. Han, S.; Pool, J.; Tran, J.; Dally, W. Learning both weights and connections for efficient neural network. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; Volume 28.
4. Romero, A.; Ballas, N.; Kahou, S.E.; Chassang, A.; Gatta, C.; Bengio, Y. Fitnets: Hints for thin deep nets. *arXiv* **2014**, arXiv:1412.6550.
5. Aguilar, G.; Ling, Y.; Zhang, Y.; Yao, B.; Fan, X.; Guo, C. Knowledge distillation from internal representations. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 7350–7357.
6. Ryu, M.; Lee, G.; Lee, K. Knowledge distillation for bert unsupervised domain adaptation. *Knowl. Inf. Syst.* **2022**, *64*, 3113–3128. [CrossRef]
7. Feng, L.; Qiu, M.; Li, Y.; Zheng, H.T.; Shen, Y. Learning to augment for data-scarce domain bert knowledge distillation. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021; Volume 35, pp. 7422–7430.

8. Wu, Y.; Rezagholizadeh, M.; Ghaddar, A.; Haidar, M.A.; Ghodsi, A. Universal-KD: Attention-based output-grounded intermediate layer knowledge distillation. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Punta Cana, Dominican Republic, 7–11 Novembe 2021; pp. 7649–7661.
9. Kim, J.; Park, J.H.; Lee, M.; Mok, W.L.; Choi, J.Y.; Lee, S. Tutoring Helps Students Learn Better: Improving Knowledge Distillation for BERT with Tutor Network. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Abu Dhabi, United Arab Emirates, 7–11 December 2022; pp. 7371–7382.
10. Li, M.; Zhao, H.; Gu, T.; Ying, D.; Liao, B. Class imbalance mitigation: A select-then-extract learning framework for emotion-cause pair extraction. *Expert Syst. Appl.* **2024**, *236*, 121386. [CrossRef]
11. Huo, Y.; Wong, D.F.; Ni, L.M.; Chao, L.S.; Zhang, J. Knowledge modeling via contextualized representations for LSTM-based personalized exercise recommendation. *Inf. Sci.* **2020**, *523*, 266–278. [CrossRef]
12. Otmakhova, J.; Verspoor, K.; Lau, J.H. Cross-linguistic comparison of linguistic feature encoding in BERT models for typologically different languages. In Proceedings of the 4th Workshop on Research in Computational Linguistic Typology and Multilingual NLP, Seattle, WA, USA, 14 July 2022; pp. 27–35.
13. Chen, Q.; Du, J.; Allot, A.; Lu, Z. LitMC-BERT: Transformer-based multi-label classification of biomedical literature with an application on COVID-19 literature curation. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2022**, *19*, 2584–2595. [CrossRef]
14. Shen, L.; Wu, Z.; Gong, W.; Hao, H.; Bai, Y.; Wu, H.; Wu, X.; Bian, J.; Xiong, H.; Yu, D.; et al. SE-MoE: A Scalable and Efficient Mixture-of-Experts Distributed Training and Inference System. *arXiv* **2023**, arXiv: 2205.10034.
15. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [CrossRef]
16. Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; Bowman, S.R. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv* **2018**, arXiv:1804.07461.
17. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving language understanding by generative pre-training. *arXiv* **2018**, arXiv:1909.11942.
18. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. Albert: A lite bert for self-supervised learning of language representations. *arXiv* **2019**, arXiv:1909.11942.
19. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
20. Sun, S.; Cheng, Y.; Gan, Z.; Liu, J. Patient knowledge distillation for bert model compression. *arXiv* **2019**, arXiv:1908.09355.
21. Sanh, V.; Debut, L.; Chaumond, J.; Wolf, T. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv* **2019**, arXiv:1910.01108.
22. Adhikari, A.; Ram, A.; Tang, R.; Lin, J. Docbert: Bert for document classification. *arXiv* **2019**, arXiv:1904.08398.
23. Sun, Z.; Yu, H.; Song, X.; Liu, R.; Yang, Y.; Zhou, D. Mobilebert: A compact task-agnostic bert for resource-limited devices. *arXiv* **2020**, arXiv:2004.02984.
24. Jiao, X.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X.; Li, L.; Wang, F.; Liu, Q. Tinybert: Distilling bert for natural language understanding. *arXiv* **2019**, arXiv:1909.10351.
25. Khattab, O.; Zaharia, M. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual, 25–30 July 2020; pp. 39–48.
26. Wang, W.; Wei, F.; Dong, L.; Bao, H.; Yang, N.; Zhou, M. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In Proceedings of the Advances in Neural Information Processing Systems, Online, 6–12 December 2020; Volume 33, pp. 5776–5788.
27. Liu, W.; Zhou, P.; Zhao, Z.; Wang, Z.; Deng, H.; Ju, Q. Fastbert: A self-distilling bert with adaptive inference time. *arXiv* **2020**, arXiv:2004.02178.
28. Zuo, S.; Zhang, Q.; Liang, C.; He, P.; Zhao, T.; Chen, W. Moebert: From bert to mixture-of-experts via importance-guided adaptation. *arXiv* **2022**, arXiv:2204.07675.
29. Kudugunta, S.; Huang, Y.; Bapna, A.; Krikun, M.; Lepikhin, D.; Luong, M.T.; Firat, O. Beyond Distillation: Task-level Mixture-of-Experts for Efficient Inference. *arXiv* **2021**, arXiv: 2110.03742.
30. Lepikhin, D.; Lee, H.; Xu, Y.; Chen, D.; Firat, O.; Huang, Y.; Krikun, M.; Shazeer, N.; Chen, Z. GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding. *arXiv* **2020**, arXiv: 2006.16668.
31. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. *arXiv* **2015**, arXiv:1503.02531.