EarlyBERT: Efficient BERT Training via Early-bird Lottery Tickets

Xiaohan Chen[1]* **Yu Cheng**[2] **Shuohang Wang**[2] **Zhe Gan**[2]
Zhangyang Wang[1] **Jingjing Liu**[2]

[1]University of Texas at Austin, [2]Microsoft Corporation
{xiaohan.chen, atlaswang}@utexas.edu
{yu.cheng, shuowa, zhe.gan, jingjl}@microsoft.com

Abstract

Heavily overparameterized language models such as BERT, XLNet and T5 have achieved impressive success in many NLP tasks. However, their high model complexity requires enormous computation resources and extremely long training time for both pre-training and fine-tuning. Many works have studied model compression on large NLP models, but only focusing on reducing inference time while still requiring an expensive training process. Other works use extremely large batch sizes to shorten the pre-training time, at the expense of higher computational resource demands. In this paper, inspired by the *Early-Bird Lottery Tickets* recently studied for computer vision tasks, we propose EarlyBERT, a general computationally-efficient training algorithm applicable to both pre-training and fine-tuning of large-scale language models. By slimming the self-attention and fully-connected sub-layers inside a transformer, we are the first to identify *structured* winning tickets in the early stage of BERT training. We apply those tickets towards efficient BERT training, and conduct comprehensive pre-training and fine-tuning experiments on GLUE and SQuAD downstream tasks. Our results show that EarlyBERT achieves comparable performance to standard BERT, with 35∼45% less training time. Code is available at https://github.com/VITA-Group/EarlyBERT.

1 Introduction

Large-scale pre-trained language models (*e.g.,* BERT (Devlin et al., 2018), XLNet (Yang et al., 2019), T5 (Raffel et al., 2019)) have significantly advanced the state of the art in the NLP field. Despite impressive empirical success, their computational inefficiency has become an acute drawback in practice. As more transformer layers are stacked with larger self-attention blocks, model complexity increases rapidly. For example, compared to BERT-Large model with 340 million parameters, T5 has more than 10 billion to learn. Such high model complexity calls for expensive computational resources and extremely long training time.

Model compression is one approach to alleviating this issue. Recently, many methods have been proposed to encode large NLP models compactly (Sanh et al., 2019; Jiao et al., 2019; Sun et al., 2019, 2020b). However, the focus is solely on reducing inference time or resource costs, leaving the process of searching for the right compact model ever more costly. Furthermore, most model compression methods start with a large pre-trained model, which may not be available in practice. Recent work (You et al., 2020b) proposes to use large training batches, which significantly shortens pre-training time of BERT-Large model but demands daunting computing resources (1,024 TPUv3 chips).

In contrast, our quest is to find a general resource-*efficient* training algorithm for large NLP models, which can be applied to both pre-training and fine-tuning stages. Our goal is to trim down the training time and avoid more costs of the total training resources (*e.g.*, taking large-batch or distributed training). To meet this challenge demand, we draw inspirations from recent work (You et al., 2020a) that explores the use of Lottery Ticket Hypothesis (LTH) for efficient training of computer vision models. LTH was first proposed in Frankle and Carbin (2019) as an exploration to understand the training process of deep networks. The original LTH substantiates a trainable sparse sub-network at initialization, but it cannot be directly utilized for efficient training, since the subnetwork itself has to be searched through a tedious iterative process. In addition, most LTH works discussed only unstructured sparsity. The study of You et al. (2020a)

*Work was done when the author interned at Microsoft.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 2195–2207
August 1–6, 2021.

presents discoveries that structured lottery tickets can emerge in early stage of training (*i.e.*, Early-Bird Ticket), and therefore a structurally sparse sub-network can be identified with much lower costs, leading to practical efficient training algorithms.

Inspired by the success of LTH and Early-Bird Ticket, we propose EarlyBERT, a general efficient training algorithm based on structured Early-Bird Tickets. Due to the vast differences between the architectures and building blocks of computer vision models and BERT, directly extending the method of You et al. (2020a) does not apply to our work. By instead using network slimming (Liu et al., 2017) on the self-attention and fully-connected sub-layers inside a transformer, we are the first to introduce an effective approach that can identify *structured winning tickets in the early stage of BERT training*, that are successfully applied for efficient language modeling pre-training and fine-tuning. Extensive experiments on BERT demonstrate that EarlyBERT can save 35∼45% training time with minimal performance degradation, when evaluated on GLUE and SQuAD benchmarks.

2 Related Work

Efficient NLP Models It is well believed that BERT and other large NLP models are considerably overparameterized (McCarley, 2019; Sun et al., 2019). This explains the emergence of many model compression works, which can be roughly categorized into quantization (Shen et al., 2020; Zafrir et al., 2019), knowledge distillation (Sun et al., 2019; Jiao et al., 2019; Sanh et al., 2019; Sun et al., 2020a,b), dynamic routing (Fan et al., 2019; Xin et al., 2020), and pruning (Li et al., 2020; Wang et al., 2019; McCarley, 2019; Michel et al., 2019). Almost all model compression methods focus on reducing inference time, while their common drawback is the reliance on fully-trained and heavily-engineered dense models, before proceeding to their compact, sparse versions - which essentially transplants the resource burden from the inference to the training stage.

Pruning is the mainstream approach for compressing BERT so far (Gordon et al., 2020). McCarley (2019) proposed to greedily and iteratively prune away attention heads contributing less to the model. Wang et al. (2019) proposed to structurally prune BERT models using low-rank factorization and augmented Lagrangian ℓ_0 norm regularization. McCarley (2019) pruned less important self-attention heads and slices of MLP layers by applying ℓ_0 regularization to the coefficient corresponding to each head/MLP layer. Others aim to reduce the training time of transformer-based models via large-batch training and GPU model parallelism (You et al., 2020b; Shoeybi et al., 2019). Our work is orthogonal to these works, and can be readily combined for further efficiency boost.

Lottery Ticket Hypothesis in Computer Vision Lottery Ticket Hypothesis (LTH) was firstly proposed in Frankle and Carbin (2019), which shed light on the existence of sparse sub-networks (*i.e.*, winning tickets) at initialization with non-trivial sparsity ratio that can achieve almost the same performance (compared to the full model) when trained alone. The winning tickets are identified by pruning fully trained networks using the so-called Iterative Magnitude-based Pruning (IMP). However, IMP is expensive due to its iterative nature. Moreover, IMP leads to unstructured sparsity, which is known to be insufficient in reducing training cost or accelerating training speed practically. These barriers prevent LTH from becoming immediately helpful towards efficient training.

Morcos et al. (2019) studies the transferability of winning tickets between datasets and optimizers. Zhou et al. (2019) investigates different components in LTH and observes the existence of super-masks in winning tickets. Lately, You et al. (2020a) pioneers to identify Early-Bird Tickets, which emerge at the early stage of the training process, and contain structured sparsity when pruned with Network Slimming (Liu et al., 2017) which adopts channel pruning. Early-bird tickets mitigate the two limitations of IMP aforementioned, and renders it possible to training deep models efficiently, by drawing tickets early in the training and then focusing on training this compact subnetwork only. Chen et al. (2021) reveals the benefit of LTH in data-efficient training, but their focus is not on saving training resources.

Lottery Ticket Hypothesis in NLP All above works evaluate their methods on computer vision models. For NLP models, previous work has also found that matching subnetworks exist in transformers and LSTMs (Yu et al., 2019; Renda et al., 2020). Evci et al. (2020) derived an algorithm for training sparse neural networks according to LTH and applied it to character-level language modeling on WikiText-103. For BERT models, a latest work (Chen et al., 2020b) found that the pre-

trained BERT models contain sparse subnetworks, found by unstructured IMP at 40% to 90% sparsity, that are independently trainable and transferable to a range of downstream tasks with no performance degradation. Their follow-up work (Chen et al., 2020a; Gan et al., 2021) pointed out similar phenomenons in pre-trained computer vision and vision-language models. Another work (Prasanna et al., 2020) aims to find structurally sparse lottery tickets for BERT, by pruning entire attention heads and MLP layers. Their experiments turn out that all subnetworks ("good" and "bad") have "comparable performance" when fined-tuned on downstream tasks, leading to their "all tickets are winning" conclusion.

Nevertheless, both relevant works (Chen et al., 2020b; Prasanna et al., 2020) examine only the pre-trained BERT model, *i.e.*, finding tickets with regard to the fine-tuning stage on downstream tasks. To our best knowledge, no existing study analyzes the LTH at the pre-training stage of BERT; nor has any work discussed efficient BERT training using LTH, for either pre-training or fine-tuning. Our work makes the first attempt of introducing LTH to both efficient pre-training and efficient fine-tuning of BERT. Our results also provide positive evidence that LTH and Early-Bird Tickets in NLP models are amendable to structured pruning.

3 The EarlyBERT Framework

In this section, we first revisit the original Lottery Ticket Hypothesis (LTH) (Frankle and Carbin, 2019) and its variant Early-Bird Ticket (You et al., 2020a), then describe our proposed EarlyBERT.

3.1 Revisiting Lottery Ticket Hypothesis

Denote $f(x;\theta)$ as a deep network parameterized by θ and x as its input. A sub-network of f can be characterized by a binary mask m, which has exactly the same dimension as θ. When applying the mask m to the network, we obtain the sub-network $f(x;\theta \odot m)$, where $\odot$ is the Hadamard product operator. LTH states that, for a network initialized with θ_0, an algorithm called Iterative Magnitude Pruning (IMP) can identify a mask m such that the sub-network $f(x;\theta_0 \odot m)$ can be trained to have no worse performance than the full model f following the same training protocol. Such a sub-network $f(x;\theta_0 \odot m)$, including both the mask m and initial parameters θ_0, is called a *winning ticket*. The IMP algorithm works as follows: (1) initialize m as an all-one mask; (2) fully train $f(x;\theta_0 \odot m)$ to obtain a well-trained θ; (3) remove a small portion of weights with the smallest magnitudes from $\theta \odot m$ and update m; (4) repeat (2)-(3) until a certain sparsity ratio is achieved.

Two obstacles prevent LTH from being directly applied to efficient training. First, the iterative process in IMP is essential to preserve the performance of LTH; however, this is computationally expensive, especially when the number of iterations is high. Second, the original LTH does not pursue any structured sparsity in the winning tickets. In practice, unstructured sparsity is difficult to be utilized for computation acceleration even when the sparsity ratio is high (Wen et al., 2016).

To mitigate these gaps, Early-Bird Tickets are proposed by You et al. (2020a), who discovers that when using structured mask m and a properly selected learning rate, the mask m quickly converges and the corresponding mask emerges as the winning ticket in the early stage of training. The early emergence of winning tickets and the structured sparsity are both helpful in reducing computational cost in the training that follows. You et al. (2020a) focuses on computer vision tasks with convolutional networks such as VGG (Simonyan and Zisserman, 2014) and ResNet (He et al., 2016). Inspired by this, we set out to explore whether there are structured winning tickets in the early stage of BERT training that can significantly accelerate language model pre-training and fine-tuning.

3.2 Discovering EarlyBERT

The proposed EarlyBERT[1] training framework consists of three steps: (i) *Searching Stage*: jointly train BERT and the sparsity-inducing coefficients to be used to draw the winning ticket; (ii) *Ticket-drawing Stage*: draw the winning ticket using the learned coefficients; and (iii) *Efficient-training Stage*: train EarlyBERT for pre-training or downstream fine-tuning.

Searching Stage To search for the key substructure in BERT, we follow the main idea of Network Slimming (NS) (Liu et al., 2017). However, pruning in NS is based on the scaling factor γ in batch normalization, which is not used in most NLP models such as BERT. Therefore, we make

[1]EarlyBERT refers to the winning ticket discovered by the proposed 3-stage framework, which is equivalent to the resulting pruned BERT model after drawing the winning ticket. We also interchangeably use EarlyBERT as the name of the proposed framework.

necessary modifications to the original NS so that it can be adapted to pruning BERT. Specifically, we propose to associate attention heads and intermediate layers of the fully-connected sub-layers in a transformer with learnable coefficients, which will be jointly trained with BERT but with an additional ℓ_1 regularization to promote sparsity.

Some studies (Michel et al., 2019; Voita et al., 2019) find that the multi-head self-attention module of transformer can be redundant, presenting the possibility of pruning some heads from each layer of BERT without hurting model capacity. A multi-head attention module (Vaswani et al., 2017) is formulated as:

$$\begin{aligned}\text{MultiHead}(Q,K,V) &= \text{Concat}(\text{h}_1,\dots,\text{h}_n)W^O \\ \text{h}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V),\end{aligned} \quad (1)$$

where n is the number of heads, and the projections W^O, W_i^Q, W_i^K, W_i^V are used for output, query, key and value. Inspired by Liu et al. (2017), we introduce a set of scalar coefficients c_i^h (i is the index of attention heads and h means "head") inside h_i:

$$\text{h}_i = c_i^h \cdot \text{Attention}(QW_i^Q, KW_i^K, VW_i^V). \quad (2)$$

After the self-attention sub-layer in each transformer layer, the output $\text{MultiHead}(Q,K,V)$ will be fed into a two-layer fully-connected network, in which the first layer increases the dimension of the embedding by 4 times and then reduces it back to the hidden size (768 for $\text{BERT}_{\text{BASE}}$ and 1,024 for $\text{BERT}_{\text{LARGE}}$). We multiply learnable coefficients to the intermediate neurons:

$$\text{FFN}(x) = c^f \cdot \max(0, xW_1 + b_1)W_2 + b_2. \quad (3)$$

These modifications allow us to jointly train BERT with the coefficients, using the following loss:

$$\mathcal{L}(f(\cdot;\theta), c) = \mathcal{L}_0(f(\cdot;\theta), c) + \lambda\|c\|_1, \quad (4)$$

where $\mathcal{L}_0$ is the original loss function used in pre-training or fine-tuning, c is the concatenation of all the coefficients in the model including those for attention heads and intermediate neurons, and λ is the hyper-parameter that controls the strength of regularization.

Note that in this step, the joint training of BERT and the coefficients are still as expensive as normal BERT training. However, the *winning strategy* of EarlyBERT is that we only need to perform this joint training for a few steps, before the winning ticket emerges, which is much shorter than the full training process of pre-training or fine-tuning. In other words, we can identify the winning tickets at a very low cost compared to the full training. Then, we draw the ticket (*i.e.*, the EarlyBERT), reset the parameters and train EarlyBERT that is computationally efficient thanks to its structured sparsity. Next, we introduce how we draw EarlyBERT from the learned coefficients.

Ticket-drawing Stage After training BERT and coefficients c jointly, we draw EarlyBERT using the learned coefficients with a magnitude-based metric. Note that we prune attention heads and intermediate neurons separately, as they play different roles.

We prune the attention heads whose coefficients have the smallest magnitudes, and remove these heads from the computation graph. We also prune the rows in W^O (see Eqn. (1)) that correspond to the removed heads. Note that this presents a design choice: should we prune the heads *globally* or *layer-wisely*? In this paper, we use layer-wise pruning for attention heads, because the number of heads in each layer is very small (12 for $\text{BERT}_{\text{BASE}}$ and 16 for $\text{BERT}_{\text{LARGE}}$). We observe empirically that if pruned globally, the attention heads in some layers may be completely removed, making the network un-trainable. Furthermore, Ramsauer et al. (2020) observes that attention heads in different layers exhibit different behaviors. This also motivates us to only compare importance of attention heads within each layer.

Similar to pruning attention heads, we prune intermediate neurons in the fully-connected sub-layers. Pruning neurons is equivalent to reducing the size of intermediate layers, which leads to a reduced size of the weight matrices W_1 and W_2 in Eqn. (3). Between global and layer-wise pruning, empirical analysis shows that global pruning works better. We also observe that our algorithm naturally prunes more neurons for the later layers than earlier ones, which coincides with many pruning works on vision tasks. We leave the analysis of this phenomenon as future work.

Efficient-training Stage We then train EarlyBERT that we have drawn for pre-training or fine-tuning depending on the target task. If we apply EarlyBERT to pre-training, the initialization θ_0 of BERT will be a random initialization, the same setting as the original LTH (Frankle and Carbin, 2019) and Early-Bird Tickets (You et al., 2020a).

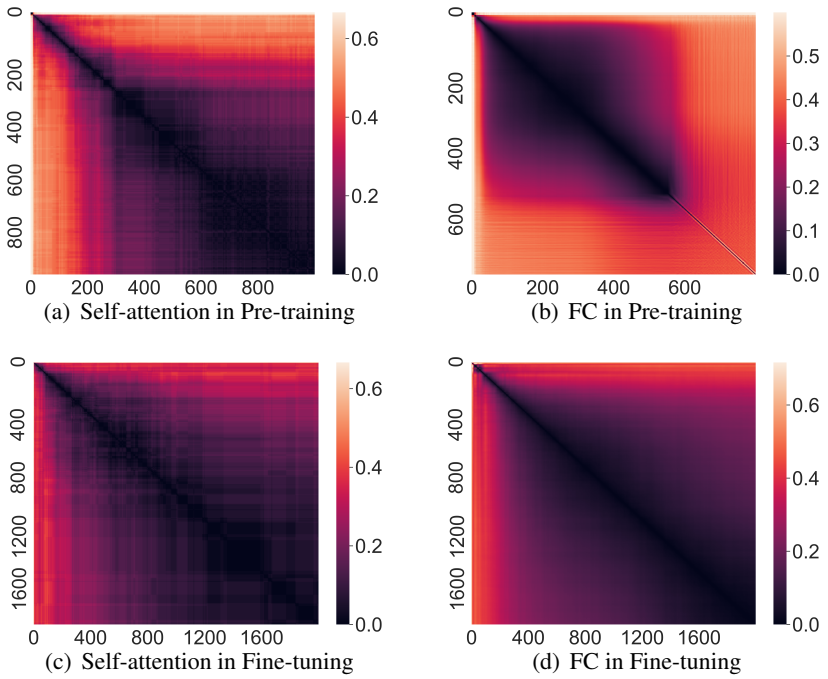

Figure 1: Illustration of mask difference in Hamming distance. Top: mask distance observed in pre-training. Bottom: mask distance observed in fine-tuning. The color represents the normalized mask distance between different training steps. The darker the color, the smaller the mask distance. In both cases, the mask converges quickly, which indicates the early emergence of the tickets.

If we apply EarlyBERT to fine-tuning, then θ_0 can be any pre-trained model. We can also moderately reduce the training steps in this stage without sacrificing performance, which is empirically supported by the findings in Frankle and Carbin (2019); You et al. (2020a) that the winning tickets can be trained more effectively than the full model. In practice, the learning rate can also be increased to speed up training, in addition to reducing training steps.

Different from unstructured pruning used in LTH and many other compression works (Frankle and Carbin, 2019; Chen et al., 2020b), structurally pruning attention heads and intermediate neurons in fully-connected layers can directly reduce the number of computations required in the transformer layer, and shrink the matrix size of the corresponding operations, yielding a direct reduction in computation and memory costs.

3.3 Validation of EarlyBERT

Early Emergence Following a similar manner in You et al. (2020a), we visualize the normalized mask distance between different training steps, to validate the early emergence of winning tickets. In Figure 1, the axes in the plots are the number of training steps finished. We only use one fully-connected sub-layer to plot Figure 1(b),1(d) due to high dimensionality. In both pre-training and fine-tuning, the mask converges in a very early stage of the whole training process. Although we observe an increase of mask distance in fully-connected layers during pre-training (in Figure 1(b)), this can be easily eliminated by early stopping and using mask distance as the exit criterion. An ablation study on how early stopping influences the performance of EarlyBERT is presented in Sec. 4.2.

Methods	MNLI	QNLI	QQP	SST-2
$\text{BERT}_{\text{BASE}}$	83.16	90.59	90.34	91.70
$\text{EarlyBERT}_{\text{BASE}}$	83.58	90.33	90.41	92.09
Random	82.26	88.87	0.12	91.17
Methods	CoLA	RTE	MRPC	
$\text{BERT}_{\text{BASE}}$	0.535	65.70	80.96	
$\text{EarlyBERT}_{\text{BASE}}$	0.527	66.19	81.54	
Random	0.514	63.86	78.57	

Table 1: Comparison between randomly-pruned models and EarlyBERT on GLUE tasks. Different from experiments in Sec. 4, here we prune only 4 heads in each layer and no intermediate neurons.

Non-trivial Sub-network Here, by *non-trivial* we mean that with the same sparsity ratio as in EarlyBERT, randomly pruned model suffers from significant performance drop. The performance drop happens even if we only prune attention heads. We verify this by running fine-tuning experiments

Methods	MNLI	QNLI	QQP	SST-2	SQuAD	Time Saved[2]
$\text{BERT}_{\text{BASE}}$	83.16	90.59	90.34	91.70	87.50	-
$\text{EarlyBERT}_{\text{BASE}}$	81.81	89.18	90.06	90.71	86.13	40∼45%
$\text{Random}_{\text{BASE}}$	79.92	84.46	89.42	89.68	84.47	45∼50%
LayerDrop (Fan et al., 2019)	81.27	88.91	88.06	89.89	84.25	∼33%
$\text{BERT}_{\text{LARGE}}$	86.59	92.29	91.59	92.21	90.76	-
$\text{EarlyBERT}_{\text{LARGE}}$	85.13	89.22	90.64	90.94	89.45	35∼40%
$\text{Random}_{\text{LARGE}}$	78.45	84.46	89.89	88.65	88.79	40∼45%
LayerDrop (Fan et al., 2019)	85.12	91.12	88.88	89.97	89.44	∼33%

Table 2: Performance of EarlyBERT (fine-tuning) compared with different baselines. We follow the official implementation of LayerDrop method (Fan et al., 2019). The protocol that we follow for measuring the training time savings is described in Sec. 4.1. We only evaluate models on large downstream tasks since our goal is improving training efficiency.

on $\text{BERT}_{\text{BASE}}$. Specifically, we prune 4 heads from each transformer layer in $\text{BERT}_{\text{BASE}}$ and EarlyBERT. We fine-tune $\text{BERT}_{\text{BASE}}$ for 3 epochs with an initial learning rate 2×10^{-5}. We run the searching stage for 0.2 epochs with $\lambda = 1 \times 10^{-4}$, draw EarlyBERT with pruning ratio $\rho = 1/3$, and then fine-tune EarlyBERT for 2 epochs with doubled initial learning rate. For the randomly pruned models, we randomly prune 4 heads in each layer and follow the same fine-tuning protocol as EarlyBERT. The reported results of randomly pruned models are the average of 5 trials with different seeds for pruning. The results on four tasks from GLUE benchmark (Wang et al., 2018) presented in Table 1 show that randomly pruned model consistently underperforms EarlyBERT with a significant gap, supporting our claim that EarlyBERT indeed identifies non-trivial sub-structures.

4 Experiments

4.1 Experimental Setting

Backbone Models Following the official BERT implementation (Devlin et al., 2018; Wolf et al., 2019), we use both $\text{BERT}_{\text{BASE}}$ (12 transformer layers, hidden size 768, 3,072 intermediate neurons, 12 self-attention heads per layer, 110M parameters in total) and $\text{BERT}_{\text{LARGE}}$ (24 transformer layers, hidden size 1,024, 4,096 intermediate neurons, 16 self-attention heads per layer, 340M parameters in total) for experiments.

Datasets We use English Wikipedia (2,500M words) as the pre-training data. For fine-tuning experiments and evaluation of models in the pre-training experiments, we use tasks from GLUE benchmark (Wang et al., 2018) and a question-answering dataset SQuAD v1.1 (Rajpurkar et al., 2016). Note that as our goal is efficient pre-training and fine-tuning, we focus on larger datasets from GLUE (MNLI, QNLI, QQP and SST-2), as it is less meaningful to discuss efficient training on very small datasets. We use the default training settings for pre-training and fine-tuning on both models. To evaluate model performance, we use Matthew's correlation score for CoLA, matched accuracy for MNLI, F1-score for SQuAD v1.1, and accuracy in percentage for other tasks on GLUE. We omit % symbols in all the tables on accuracy results.

Implementation Details For the vanilla BERT, we fine-tune on GLUE datasets for 3 epochs with initial learning rate 2×10^{-5}, and for 2 epochs on SQuAD with initial learning rate 3×10^{-5}; we use AdamW (Loshchilov and Hutter, 2017) optimizer for both cases. For pre-training, we adopt LAMB optimization technique (You et al., 2020b), which involves two phases of training: the first 9/10 of the total training steps uses a sequence length of 128, while the last 1/10 uses a sequence length of 512. Pre-training by default has 8,601 training steps and uses 64k/32k batch sizes and $6 \times 10^{-3}/4 \times 10^{-3}$ initial learning rates for the two phases, respectively. All experiments are run on 16 NVIDIA V100 GPUs.

Training Time Measuring Protocol We strictly measure the training time saving of EarlyBERT on the QQP task in GLUE using CUDA benchmark mode. To get rid of the influence of the hardware environment at our best, we individually measure the time elapsed during each step and calculate the average time per step over the whole training process. The time for data I/O is excluded. The training time of EarlyBERT includes both the searching stage and the efficient training stage.

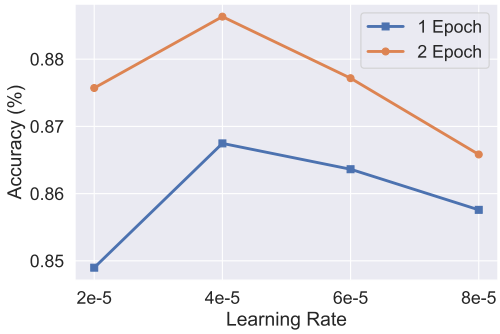

Figure 2: Effect of reducing training epochs and up-scaling learning rate for EarlyBERT in fine-tuning. The combination of 2-epoch fine-tuning and 4×10^{-5} turns out to be the optimal choice.

4.2 Experiments on Fine-tuning

The main results of EarlyBERT in fine-tuning are presented in Table 2. According to the observation of the early emergence of tickets in Sec. 3.3, we run the searching stage for 0.2 epochs (which accounts for less than 7% of the cost of a standard 3-epoch fine-tuning) with $\lambda = 1 \times 10^{-4}$ for all tasks. When drawing EarlyBERT, we prune 4 heads in each layer from $\text{BERT}_{\text{BASE}}$ and 6 heads from $\text{BERT}_{\text{LARGE}}$, *and* globally prune 40% intermediate neurons in fully-connected sub-layers in both models, instead of pruning only heads as in Table 1. After this, we re-train the EarlyBERT models for reduced training epochs (from 3 to 2) on GLUE benchmark and the learning rate scaled up by 2 times to buffer the effect of reduced epochs. For SQuAD dataset, we keep the default setting, as we find SQuAD is more sensitive to the number of training steps. The selection of these hyperparameters are based on the ablation studies that follow the main results in Table 2, in which we investigate the effects of the number of training epochs, learning rate during downstream fine-tuning, the regularization strength λ, and the pruning ratios on self-attention heads and intermediate neurons.

Several observations can be drawn from Table 2. First, in most tasks, EarlyBERT saves over 40% of the total training time with 4 self-attention heads pruned in each layer and 40% FC neurons pruned globally, without inducing much performance degradation. Specifically, following the training time measurement protocol in Sec. 4.1, we observe that EarlyBERT saves 42.97% of the total training time of a full BERT model on QQP task. The time saving slightly differs over various tasks, hence we report a range of saving time. Here, $\text{Random}_{\text{BASE}}$ saves slightly more training time because random pruning skips the searching stage in $\text{EarlyBERT}_{\text{BASE}}$, but it induces much more accuracy drop. $\text{EarlyBERT}_{\text{BASE}}$ can also outperform another strong baseline LayerDrop (Fan et al., 2019), which drops one third of the layers so that the number of remaining parameters are comparable to ours. Note that LayerDrop models are fine-tuned for three full epochs, yet EarlyBERT is still competitive in most cases. Second, we consistently observe obvious performance advantage of EarlyBERT over randomly pruned models, which provides another strong evidence that EarlyBERT does discover nontrivial key sparse structures. Even though there still exists a margin between EarlyBERT and the baseline (You et al. (2020a) also observed similar phenomenon in their tasks), the existence of structured winning tickets and its potential for efficient training is highly promising. We leave as future work to discover winning tickets of higher sparsity but better quality.

λ	10^{-4}	10^{-3}	10^{-2}
	88.55	88.43	88.42
# Pruned Heads	4	5	6
Layer-wise pruning	**88.55**	88.13	87.65
# Pruned Neurons	30%	40%	50%
Layer-wise pruning	88.18	**88.22**	87.90
Global pruning	**88.31**	88.23	87.91

Table 3: Ablation of regularization strength λ and pruning ratios on self-attention heads and intermediate neurons. All numbers are the average of three runs with different random seeds on four tasks on GLUE (MNLI/QNLI/QQP/SST-2).

Ablation Studies on Fine-tuning We perform extensive ablation studies to investigate important hyper-parameter settings in EarlyBERT, using $\text{EarlyBERT}_{\text{BASE}}$ as our testing bed. For all experiments, we use the average accuracy on the larger datasets from GLUE benchmark (MNLI, QNLI, QQP and SST-2) as the evaluation metric.

- **Number of training epochs and learning rate.** We first investigate whether we can properly reduce the number of training epochs, and if scaling the learning rate can help compliment the negative effect caused by reducing training steps. Results in Figure 2 show that when we fine-tune EarlyBERT for fewer epochs on GLUE, up-scaling learning rate first helps to recover performance, and then causes decrease again. We will use two epochs and 4×10^{-5} as learning rate

Methods	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST-2	SQuAD
$\text{BERT}_{\text{BASE}}$	0.45	81.40	84.07	89.86	89.80	60.29	90.48	87.60
$\text{EarlyBERT}_{\text{BASE}}$	0.41	79.97	80.39	89.86	89.44	61.01	90.94	85.48
$\text{BERT}_{\text{LARGE}}$	0.50	83.56	85.90	90.44	90.45	59.93	92.55	90.43
$\text{EarlyBERT}_{\text{LARGE}}$	0.47	82.54	85.54	90.46	90.38	61.73	91.51	89.36

Table 4: Performance of EarlyBERT (pre-training) compared with BERT baselines. Different from fine-tuning experiments, we evaluate pre-trained models on all downstream tasks in GLUE and SQuAD since fine-tuning cost is negligible compared to the dominant pre-training cost.

for EarlyBERT on GLUE experiments.

- **Regularization strength λ.** A proper selection of the regularization strength λ decides the quality of the winning ticket, consequently the performance of EarlyBERT after pre-training/fine-tuning. Results in Table 3 show that λ has marginal influence on EarlyBERT performance. We use $\lambda = 10^{-4}$ that achieves the best performance in following experiments.

- **Pruning ratios ρ.** We further investigate the effects of different pruning ratios as well as layer-wise/global pruning on the performance of EarlyBERT. As discussed in Sec. 3.2, we only consider layer-wise pruning for self-attention heads. Table 3 shows that the performance monotonically decreases when we prune more self-attention heads from BERT; however, we see a slight increase and then a sharp decrease in accuracy, when the pruning ratio is raised for intermediate neurons in fully-connected sub-layers (40% pruning ratio seems to be the sweet spot). We also observe consistent superiority of global pruning over layer-wise pruning for intermediate neurons.

- **Early-stop strategy for searching.** In Figure 1, we show the early emergence of winning tickets in BERT when trained with ℓ_1 regularization, suggesting we stop the searching stage early to save computation while still generating high-quality tickets. Here, we study how the early-stop strategy influences the model performance. We fine-tune EarlyBERT on QNLI following the same setting described earlier in this section, but stop the searching stage at different time points during the first epoch for searching. Results in Figure 3 show (i) an abrupt increase in accuracy when we stop at 20% of the first epoch; (ii) slight increase when we delay the stop till the end of the first epoch. Considering training efficiency, we think 20$\sim$40% makes suitable stopping time.

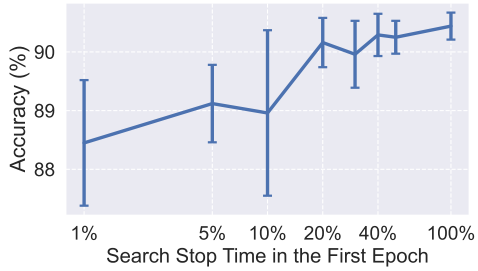

Figure 3: How various time points of early stopping for searching influences EarlyBERT performance.

Time Saving / Prune Ratio	3 Heads	4 Heads	5 Heads	6 Heads
FC - 30%	-35.78%	-38.66%	-41.26%	-45.34%
	89.62%	89.55%	89.60%	89.50%
FC - 40%	-39.72%	-42.97%	-43.93%	-44.49%
	89.66%	89.61%	89.58%	89.38%
FC - 50%	-43.89%	-45.54%	-47.02%	-48.53%
	89.54%	89.35%	89.34%	89.31%

Table 5: Training time savings vs. accuracies of EarlyBERT on the QQP task in GLUE with different pruning ratios for self-attention heads and FC neurons.

Trade-off Between Efficiency and Performance We vary the pruning ratios for the FC layers and the number of self-attention heads pruned in each layer in EarlyBERT, fine-tune the models on QQP in GLUE, and obtain the corresponding validation accuracies and training time savings following the protocol above. Results are shown in Table 5. We can see clear correlations between the training time saving and the accuracy — the more FC neurons or self-attention heads pruned, the more training time saving yet the larger accuracy drop. Moreover, for most combinations of these two hyper-parameters, the accuracy drop is within 1%, which also supports the efficiency of EarlyBERT.

4.3 Experiments on Pre-training

We also conduct pre-training experiments and present the main results in Table 4. We run the

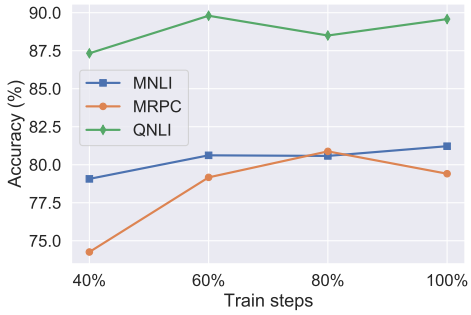

Figure 4: Effect of reducing training steps in pre-training on EarlyBERT$_{\text{BASE}}$.

search stage for 400 steps of training in the first training phase that uses a sequence length of 128 which only accounts for less than 3% of a standard pre-training, with $\lambda = 1 \times 10^{-4}$. When we draw EarlyBERT, similar to the settings in fine-tuning experiments, we prune 4 heads in each layer from BERT$_{\text{BASE}}$ and 6 heads from BERT$_{\text{LARGE}}$; however, we prune slightly fewer (30%) intermediate neurons in fully-connected sub-layers in both models, since we empirically observe that pre-training is more sensitive to aggressive intermediate neuron pruning. In both phases of pre-training, we reduce the training steps to 80% of the default setting when training EarlyBERT (based on the ablation study shown in Figure 4). Other hyper-parameters for pre-training follow the default setting described in Sec. 4.1. All models are fine-tuned and evaluated on GLUE and SQuAD v1.1 with the default setting.

Different from fine-tuning experiments, the pre-training stage dominates the training time over the downstream fine-tuning, and thus we only consider the training time saving during pre-training. Since the randomly pruned models do not have competitive performance in fine-tuning experiments as shown in Sec. 4.2, we focus on comparing EarlyBERT with the full BERT baseline.

From the results presented in Table 4, we can see that on downstream tasks with larger datasets such as QNLI, QQP and SST-2, we can achieve accuracies that are close to BERT baseline (within 1% accuracy gaps except for EarlyBERT$_{\text{BASE}}$ on MNLI and SQuAD). However, on downstream tasks with smaller datasets, the patterns are not consistent: we observe big drops on CoLA and MRPC but improvement on RTE. Overall, EarlyBERT achieves comparable performance while saving 30∼35% training time thanks to its structured sparsity and reduction in training steps.

Reducing Training Steps in Pre-training We investigate whether EarlyBERT, when non-essential heads and/or intermediate neurons are pruned, can train more efficiently, and whether we can reduce the number of training steps in pre-training. This can further help reduce training cost in addition to the efficiency gain from pruning. We use EarlyBERT$_{\text{BASE}}$-Self (only self-attention heads are pruned when drawing the winning ticket) as the testing bed. Figure 4 shows the performance decreases more when we reduce the number of training steps to 60% or less. Reducing it to 80% seems to be a sweet point with the best balance between performance and efficiency.

4.4 Comparison with Previous Lottery Tickets Work in NLP

On one hand, two relevant works (Chen et al., 2020b; Prasanna et al., 2020) only investigate lottery tickets on pre-trained NLP models for fine-tuning on the downstream tasks, while EarlyBERT makes the first attempt of introducing lottery tickets to both fine-tuning and pre-training stages, and provides empirical evidence that NLP models are amendable to structured pruning.

On the other hand, EarlyBERT pursues structured sparsity while Chen et al. (2020b) promotes unstructured sparsity, which is hardware unfriendly and provides almost no acceleration, besides the high cost of IMP. As an implicit comparison, Chen et al. (2020b) induces 0.4% accuracy drop on SQuAD v1 dataset compared to the BERT baseline with 40% unstructured sparsity (comparable with our settings in Section 4.2), while EarlyBERT induces 1.37% accuracy drop. Note that Chen et al. (2020b) uses 6x training times (because IMP reaches 40% sparsity with 6 iterations) and 4.69x FLOPs, but EarlyBERT uses only 0.76x training times and FLOPs in contrast.

5 Conclusion

In this paper, we present EarlyBERT, an efficient framework for large-scale language model pre-training and fine-tuning. Based on Lottery Ticket Hypothesis, EarlyBERT identifies structured winning tickets in an early stage, then uses the pruned network for efficient training. Experimental results demonstrate that the proposed method is able to achieve comparable performance to standard BERT with much less training time. Future work includes exploring more data-efficient strategies to enhance the current training pipeline.

References

Tianlong Chen, Yu Cheng, Zhe Gan, Jingjing Liu, and Zhangyang Wang. 2021. Ultra-data-efficient gan training: Drawing a lottery ticket first, then training it toughly. *arXiv preprint arXiv:2103.00397*.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Michael Carbin, and Zhangyang Wang. 2020a. The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. *arXiv preprint arXiv:2012.06908*.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020b. The lottery ticket hypothesis for pre-trained bert networks. In *NeurIPS*.

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

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. 2020. Rigging the lottery: Making all tickets winners. In *ICML*.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *ICLR*.

Zhe Gan, Yen-Chun Chen, Linjie Li, Tianlong Chen, Yu Cheng, Shuohang Wang, and Jingjing Liu. 2021. Playing lottery tickets with vision and language. *arXiv preprint arXiv:2104.11832*.

Mitchell A Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing bert: Studying the effects of weight pruning on transfer learning. *arXiv preprint arXiv:2002.08307*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joseph E. Gonzalez. 2020. Train large, then compress: Rethinking model size for efficient training and inference of transformers. In *ICML*.

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. 2017. Learning efficient convolutional networks through network slimming. In *ICCV*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

J Scott McCarley. 2019. Pruning a bert-based question answering model. *arXiv preprint arXiv:1910.06360*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *NeurIPS*.

Ari Morcos, Haonan Yu, Michela Paganini, and Yuandong Tian. 2019. One ticket to win them all: generalizing lottery ticket initializations across datasets and optimizers. In *NeurIPS*.

Sai Prasanna, Anna Rogers, and Anna Rumshisky. 2020. When bert plays the lottery, all tickets are winning. *arXiv preprint arXiv:2005.00561*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *EMNLP*.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, Victor Greiff, et al. 2020. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*.

Alex Renda, Jonathan Frankle, and Michael Carbin. 2020. Comparing rewinding and fine-tuning in neural network pruning. *arXiv preprint arXiv:2003.02389*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2020. Q-bert: Hessian based ultra low precision quantization of bert. In *AAAI*.

M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

S. Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. In *EMNLP*.

Siqi Sun, Zhe Gan, Yu Cheng, Yuwei Fang, Shuohang Wang, and Jingjing Liu. 2020a. Contrastive distillation on intermediate representations for language model compression. *arXiv preprint arXiv:2009.14167*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020b. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In *ACL*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *arXiv preprint arXiv:1905.09418*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2019. Structured pruning of large language models. *arXiv preprint arXiv:1910.04732*.

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2016. Learning structured sparsity in deep neural networks. In *NeurIPS*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. DeeBERT: Dynamic early exiting for accelerating BERT inference. In *ACL*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Haoran You, Chaojian Li, Pengfei Xu, Yonggan Fu, Yue Wang, Xiaohan Chen, Richard G. Baraniuk, Zhangyang Wang, and Yingyan Lin. 2020a. Drawing early-bird tickets: Toward more efficient training of deep networks. In *ICLR*.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2020b. Large batch optimization for deep learning: Training bert in 76 minutes. In *ICLR*.

Haonan Yu, Sergey Edunov, Yuandong Tian, and Ari S Morcos. 2019. Playing the lottery with rewards and multiple languages: lottery tickets in rl and nlp. In *ICLR*.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert. *arXiv preprint arXiv:1910.06188*.

Hattie Zhou, Janice Lan, Rosanne Liu, and Jason Yosinski. 2019. Deconstructing lottery tickets: Zeros, signs, and the supermask. In *NeurIPS*.

A More Comparison with BERT Baseline

For more explicit comparison, we conduct a two-way fine-tuning experiment in addition to the main results in Table 2. All results are averages of 3 runs.

We first increase the training cost of EarlyBERT to match BERT performance by extending the searching stage to a full epoch, which, according to our ablation study in Figure 3, helps to improve the performance of EarlyBERT. In this case, EarlyBERT still has 16% time and FLOPs savings, with comparable performance shown in Table 6.

Secondly, we reduce the training steps of BERT to match the FLOPs of EarlyBERT, inducing obvious gaps between BERT and EarlyBERT as presented in Table 7.

GLUE Task	MNLI	QNLI	QQP	SST-2
BERT	83.48%	90.43%	90.37%	91.86%
EarlyBERT	83.36%	90.44%	90.33%	91.55%

Table 6: We increase the number of training steps of EarlyBERT so that it achieves very close performances to the BERT baseline on the larger four tasks in GLUE benchmark.

GLUE Task	MNLI	QNLI	QQP	SST-2
BERT - Reduced	82.85%	89.86%	89.45%	91.70%
EarlyBERT	83.26%	90.16%	90.22%	91.67%

Table 7: Comparison between the performance of BERT and EarlyBERT with the same training time on the larger four tasks on GLUE benchmark by reducing the number of training steps of BERT. Obvious gaps can be observed on all four tasks but SST-2.

B Searching EarlyBERT using on the Masked Language Modeling Task

It is found in Chen et al. (2020b) selecting a winning ticket for BERT fine-tuning on the masked language modeling task (MLM), i.e., pre-training objective makes for better tickets performing on many of the downstream tasks. Here, we try the experiments of using the MLM objective during the searching stage. Results are summarized in Table 8. Our main observations include:

- When using the MLM objective for the searching stage, the mask distance for both self-attention heads and FC neurons converged well and quickly within 100 training steps.
- We first apply the global pruning method to the FC neurons because we observed better performance of EarlyBERT with that method. However, while we previously found in EarlyBERT that the latter layers will be pruned more, we observed the opposite phenomenon when using MLM objective — the former layers are pruned more instead. In terms of accuracy, we observed significant gaps compared to EarlyBERT.
- Based on the above observations, we also applied layerwise pruning for MLM experiments (shown in the last row of Table 8). We did see improved accuracy with layerwise pruning but the gaps between EarlyBERT are still large (except on QQP).

GLUE Task	MNLI	QNLI	QQP	SST-2
BERT	83.36%	90.53%	90.41%	91.61%
EarlyBERT	81.97%	88.68%	89.26%	90.48%
MLM - FC Global	78.36%	84.84%	88.86%	88.65%
MLM - FC Layerwise	79.01%	86.55%	89.16%	88.53%

Table 8: Comparison of the accuracies of EarlyBERT and EarlyBERT with winning tickets searched using MLM objective on downstream tasks in GLUE.

C The Effect of Reduced Training Steps during Pre-training

We perform the same as the analysis of the effect of reduced training steps during pre-training in Figure 4 for both the vanilla BERT and EarlyBERT. We calculate how performance will be influenced due to the reduced training steps. We use F1 score for SQuAD, Matthew's correlation score for CoLA and accuracy for all other tasks on GLUE as the metric. We report the performance reduction (or

Training Steps	BERT	EarlyBERT
100%	0.00%	0.00%
80%	-1.94%	+1.96%
60%	-2.48%	-1.42%
40%	-3.62%	-3.43%

Table 9: Effects of reduced training steps for BERT and EarlyBERT in average on GLUE benchmark tasks and SQuAD.

gain) in percentage average on all tasks, normalized by the performance of baseline, i.e., BERT or EarlyBERT trained with the default number of training steps. Similar metric is used in DistilBERT (Sanh et al., 2019). Results are shown in Table 9. We can see that using only 80% training steps actually improves the performance of EarlyBERT on average but in contrast hurts the performance of BERT. Similarly, using 60% training steps hurts BERT more than EarlyBERT. And as expected, saving more training steps generally hurt more. We think this is one piece of evidence that motivated us to use reduced training steps for EarlyBERT.