Token and Head Adaptive Transformers for Efficient Natural Language Processing

Chonghan Lee[1], **Md Fahim Faysal Khan**[1], **Rita Brugarolas Brufau**[2], **Ke Ding**[2], **Vijaykrishnan Narayanan**[1]
[1]The Pennsylvania State University
[2]Intel
[1]{cvl5361,mzk591,vxn9}@psu.edu
[2]{rita.brugarolas.brufau,ke.ding}@intel.com

Abstract

While pre-trained language models like BERT (Devlin et al., 2019) have achieved impressive results on various natural language processing tasks, deploying them on resource-restricted devices is challenging due to their intensive computational cost and memory footprint. Previous approaches mainly focused on training smaller versions of a BERT model with competitive accuracy under limited computational resources. In this paper, we extend *Length Adaptive Transformer* (Kim and Cho, 2021) and propose to design *Token and Head Adaptive Transformer*, which can compress and accelerate various BERT-based models via simple fine-tuning. We train a transformer with a progressive token and head pruning scheme, eliminating a large number of redundant tokens and attention heads in the later layers. Then, we conduct a multi-objective evolutionary search with the overall number of floating point operations (FLOPs) as its efficiency constraint to find joint token and head pruning strategies that maximize accuracy and efficiency under various computational budgets. Empirical studies show that a large portion of tokens and attention heads could be pruned while achieving superior performance compared to the baseline BERT-based models and Length Adaptive Transformers in various downstream NLP tasks. MobileBERT(Sun et al., 2020) trained with our joint token and head pruning scheme achieves a GLUE score of 83.0, which is 1.4 higher than Length Adaptive Transformer and 2.9 higher than the original model.

1 Introduction

The Transformer (Vaswani et al., 2017) has become the dominant neural architecture used in natural language processing. Especially, pre-trained models using the Transformer architecture as their backbone, e.g., BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and GPT-3 (Brown et al., 2020), have shown significant accuracy improvement across various natural language processing tasks. However, the amount of computation required for these large neural networks is enormous, which leads to a substantial increase in energy consumption and carbon footprint. For example, an article points out that in order to train a GPT-3 networks using standard NVIDIA gpus, it would roughly take 190,000 KWh of energy which is equivalent to producing around 85,000 Kg CO_2 considering America's average carbon emission intensity (Katyanna Quach, 2020). Despite their remarkable performance, these power hungry models pose a great hindrance along the pathway to sustainable systems. Hence, it is crucial to make the large-scale networks efficient and sustainable to be deployed on various hardware with different computational budgets and adopted for real-world applications.

There have been efforts to improve the efficiency of the large language models (see section 6 for a discussion in detail.) One of the effective approaches has focused on removing redundant model parameters. Recent works (Voita et al., 2019; Michel et al., 2019) show that only a small subset of heads are important and mainly attributed to the final decision making and some heads could even be pruned at test time to improve efficiency. Other works successfully improve the efficiency of BERT-based models by pruning out word token vectors. Length Adaptive Transformer (Kim and Cho, 2021) based on PoWER-BERT (Goyal et al., 2020) demonstrates that training an adaptive transformer with progressive word token pruning improves accuracy-efficiency trade-off that can satisfy any target computational budget. In this paper, we focus on developing a framework capable of training an adaptive Transformer that jointly eliminates redundant attention heads and word tokens to maximize accuracy and minimize required computational resources. We perform a multi-objective evolutionary search to find a full Pareto frontier of joint token and attention-head pruning schemes that provides optimal accuracy-efficiency trade-offs given any computational budget.

We applied our framework to BERT-based models with different sizes. Our progressive head pruning scheme combined with word token pruning allows heavy head pruning in the last layers of BERT, MobileBERT, and DistilBERT(Sanh et al., 2019) without any accuracy loss. Furthermore, the final models trained with our framework are robust to prune even more tokens along with attention heads. This results in superior accuracy and latency trade-off than simply pruning word tokens. Empirical evaluations show that MobileBERT trained with our joint token and head pruning scheme achieves a GLUE score of 83.0, which is 1.4 higher than Length Adaptive Transformer and 2.9 higher than the original model. Our adaptive BERT and DistilBERT also achieve higher accuracy with less computational cost compared to Length Adaptive Transformers. On SQuADv1.1 (Rajpurkar et al., 2016) question answering task, Token and Head Adaptive Transformers (THAT) obtain higher dev F1 scores with less computational resources compared to Length Adaptive Transformers and the original BERT-based models.

2 Background

In this section, we present the main building blocks of our framework. We review multi-head attention, the essential mechanism in the Transformer architecture. Our framework uses multi-head self-attention to decide which tokens and heads to prune in each layer. Then, we review PoWER-BERT and Length-Adaptive Transformer, which are recent works that eliminate tokens to improve efficiency in BERT-based models.

2.1 Multi-head Attention

The multi-head attention mechanism decomposes the scaled dot-product attention to extract independent features from the same input sequence in parallel. Let $\mathbf{x} = (x_1, x_2, \dots, x_T)$ be a sequence of T word token vectors where $x_t \in \mathbb{R}^d$, and $q \in \mathbb{R}^d$ be a query vector. An attention mechanism is defined as

$$Attention(\mathbf{x}, q) = W_o \sum_{t=1}^{T} \alpha_t(q) W_v x_t \qquad (1)$$

where

$$\alpha_t(q) = softmax\left(\frac{q^\top W_q^\top W_k x_t}{\sqrt{d}}\right) \qquad (2)$$

$W_o, W_v, W_q, W_k \in \mathbb{R}^{d \times d}$ are trainable weight matrices and query q comes from the same sequence $\mathbf{x}$ in self-attention. Then multi-head attention is defined as

$$MHAttention(\mathbf{x}, q) = \sum_{h=1}^{H} Attention_h(\mathbf{x}, q) \qquad (3)$$

where H is a set of attention heads h and $Attention_h$ is a decomposed low-rank attention from the head h. All the representation outputs from $Attention_h$ are created from the same input and merged together to produce a single output. Studies from (Voita et al., 2019; Michel et al., 2019) have shown possible efficiency improvement from pruning some features extracted from the heads.

2.2 PoWER-BERT

PoWER-BERT prunes out redundant word token vectors $x_t \in \mathbf{x}$ at each layer of BERT model based on the attention significance score α_t (Goyal et al., 2020). While having the same number of parameters as BERT model, PoWER-BERT significantly reduces the computational cost. PoWER-BERT requires sequence length configuration search and re-training steps. In the sequence length configuration search step, auxiliary retention parameters and a regularizer parameter are briefly introduced in the model and the loss function, respectively, to approximate the number of retained token vectors across layers under the desired accuracy and efficiency trade-off. Then the model is re-trained based on the searched sequence length configuration $\mathbf{z} = (z_1, z_2, \dots, z_N)$ where N is the total number of the encoder layers and z_n is the number of tokens to keep at layer n.

PoWER-BERT significantly improves the efficiency and achieves a better accuracy-efficiency trade-off than DistilBERT and other models with different compression techniques. However, the sequence length configuration search and re-training steps need to be repeated for each computational budget. Furthermore, PoWER-BERT's token pruning is limited to sequence-level classification tasks since it progressively prunes out hidden token vectors on each layer.

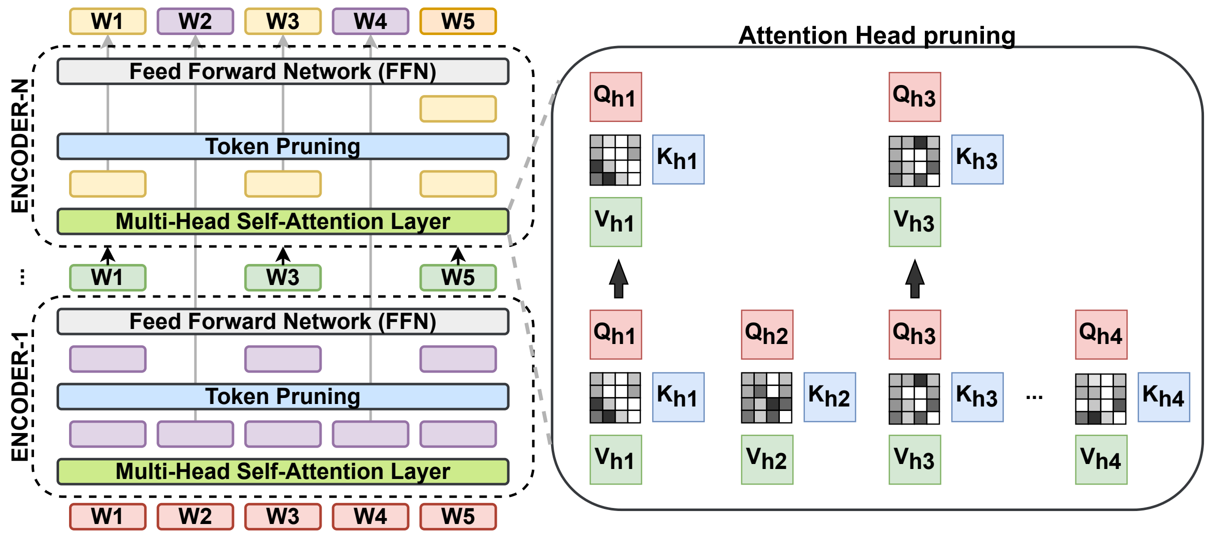

Figure 1: Overview of joint token and head pruning. The encoder blocks on the left show pruned token features as they pass to the next layer and restored at the last layer. On the right, only a subset of attention heads are aggregated to update the retained tokens to further reduce computational cost while achieving higher accuracy with better generalization.

2.3 Length Adaptive Transformer

Based on PoWER-BERT, (Kim and Cho, 2021) proposed a framework to train Length-Adaptive Transformers to reduce expensive computational cost caused by the repetitive search and re-training phases from PoWER-BERT. They train the model once with random sequence length configurations to make the model robust to various token drop during inference. Then, multi-objective evolutionary search on the adaptive model provides sequence length configurations that result in optimal trade-offs between accuracy and efficiency for any given computational budget. They also introduced drop-and-restore process to separately store the dropped token features and restore them in the final hidden layer, which makes the model applicable to token-level applications such as span-based question answering.

Our work combines findings of (Voita et al., 2019; Michel et al., 2019) and Length Adaptive Transformer to design joint token and attention head adaptive transformers which maximizes the accuracy-efficiency trade-offs.

3 Token and Head Adaptive Transformer

In this section, we describe the pruning strategy based on attention importance and head sensitivity score to remove tokens and attention heads respectively. We train Token and Head Adaptive Transformers with the pruning strategy to make the final models robust to arbitrary attention head and token drops at inference time. Fig. 1 illustrates how tokens and heads are jointly pruned in Transformer models. Then we conduct a multi-objective evolutionary search on the trained model to find the optimal pruning configuration that meets the target computational budget.

3.1 Attention score for token pruning

The importance scoring function for a sequence of tokens is based on the self-attention score.

$$I_t(x) = \alpha_t(x) = softmax\left(\frac{x^\top W_q^\top W_k x_t}{\sqrt{d}}\right) \tag{4}$$

The function measures the attention imposed by x_t on the other words $x \in \mathbf{x}$. Intuitively, if the attention score for a token x has a high value then it is likely to influence the final model decision. The significance score of x is the overall attention score aggregated over the heads. Our adaptive model dynamically prunes tokens based on the self-attention scores online.

3.2 Sensitivity score for head pruning

Following the first order method from (Michel et al., 2019; Molchanov et al., 2016), the sensitivity score is based on an unlearned gradient measure of attention heads.

$$I_h = \mathbb{E}_{x \sim X}\left|Attention_h(x)^T \frac{\partial \mathcal{L}(x)}{\partial Attention_h(x)}\right| \tag{5}$$

where X is the data distribution and $\mathcal{L}(x)$ is the loss on sample x. In our framework, we estimate the sensitivity score of each head on a training dataset and use the score as a proxy for determining which heads to prune in each layer. Without additional training cost, smaller models with a fewer number of attention heads could be efficiently sampled from our trained model using weight sharing.

3.3 Training Token and Head Adaptive Transformer

First, we measure the sensitivity score of all the attention heads by running a single forward pass of training dataset on a baseline transformer model fine-tuned for a downstream task. Then, we randomly generate a sequence length and the head configuration at each iteration.

For the head configuration, we sequentially sample the number of heads m_{i+1} at the $(i+1)$-th layer within the range $[m'_{i+1}, m_i]$. m'_{i+1} is the minimum number of retained heads in $(i+1)$-th layer set from the lower bound head pruning configuration which is progressively reducing the number of heads at a constant rate to retain only one attention head in the last layer. We set the lower bound configuration to retain a single head in the last layer based on the empirical results from (Michel et al., 2019) that found ablating all heads except one within a single layer does not significantly impact the performance. m_i is the number of retained heads in the previous layer.

For the token configuration, we followed (Kim and Cho, 2021) and sequentially sample the number of retained token n_{i+1} at the $(i+1)$-th layer within the range$[(1-p)n_i, n_i]$ where n_i is the number of retained tokens in the previous layer and p is the token drop probability. Additionally, we apply LayerDrop (Fan et al., 2019) and randomly dropout encoder layers during training to make the model robust to the random token drop.

We applied the sandwich rule training technique (Yu and Huang, 2019) to avoid difficulty in convergence as discussed in (Kim and Cho, 2021). First, we optimize the model with the upper bound configuration where none of the tokens and heads are eliminated. Then we apply in-place distillation to update the model with the lower bound with the maximum pruning configuration and with other randomly sampled intermediate pruning configurations to transfer the knowledge from the full model to the sparse models with various pruning schemes. In each iteration, the full model and the sparse model are optimized simultaneously. In practice, Token and Head Adaptive Transformer only needs to be trained for the same steps as Length Adaptive Transformer model. After training we get the superior accuracy-latency trade-off of sampled token and head pruning compared to Length Adaptive Transformer models.

3.4 Evolutionary Search of head and token pruning configurations

After training a Token and Head Adaptive Transformer, we apply evolutionary search to find the optimal token and head configurations for the model that satisfy the target computational budgets. Compared to training models specialized for every scenario, evolutionary search is computationally efficient. It only requires inference on small validation set for each pruning configuration.

We first initialize the population of joint token and head pruning configurations with constant drop ratios. The joint configurations are generated with evenly spaced token and head drop rates to have the initial population uniformly distributed between the upper bound and the lower bound pruning configurations. At each iteration, we evolve the population to consist only of configurations with the optimal accuracy-efficiency trade-offs that lie on a new Pareto frontier by mutation and crossover. A mutation alters an original pruning configuration $(j_1, \cdots, j_L)$ to $(j'_1, \cdots, j'_L)$. It involves a probability that an arbitrary element j_i for i-th layer in a pruning configuration $(j_1, \cdots, j_L)$ will be altered to a new value j'_i sampled from the uniform distribution $\left(j'_{i-1}, j_{i+1}\right)$. A crossover takes two joint configurations and averages the pruning values at each layer. In each evolution iteration, we maintain n_m joint configurations from mutation and n_c joint configurations from the crossover. The final iteration will generate the furthest Pareto frontier that consists of the joint configurations with the optimal accuracy-efficiency trade-offs.

4 Experiments

We conduct extensive experiments to evaluate our framework and compare to the baseline models and Length-Adaptive Transformers.

4.1 Model and Data

We investigate three Transformer-based models: BERT, DistilBERT, and MobileBERT. BERT is a language model with a Transformer encoder as

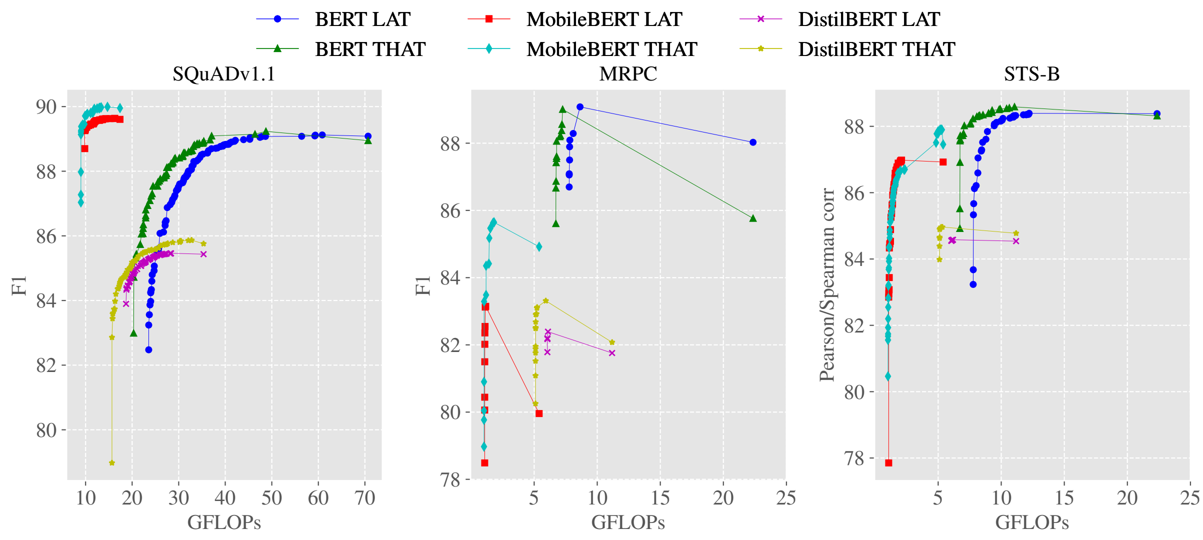

Figure 2: Pareto frontier curves of F1 score (for SQuAD v1.1 and MRPC) and Pearson/Spearman correlation score (for STS-B) to GFLOPs. We apply the proposed method to BERT, DistilBERT, and MobileBERT and compare them to the Length-Adaptive Transformers.

its base building block. The model consists of 12 encoder layers and 12 self-attention heads in each layer (144 heads in total). DistilBERT is a compact Transformer model based on BERT. Knowledge distillation is performed during pre-training to transfer knowledge from BERT teacher model to DistilBERT. DistilBERT has 6 encoder layers and 12 self-attention heads in each layer (72 heads in total). MobileBERT is a thin version of BERT$_{Large}$ that has bottleneck structures to significantly compress the model size. The model consists of 24 encoder layers and 4 self-attention heads in each layer (96 heads in total). We use the Hugging Face implementation and focus on `base-uncased` models. We use Stanford Question Answering Dataset (SQuAD v1.1) and GLUE benchmarks (Wang et al., 2018) including CoLA, STS-B, MRPC, SST-2, QNLI, QQP, and MNLI-mm to evaluate the proposed approach.

4.2 Baseline methods and evaluation metrics

We compare our approach to standard Transformer models fine-tuned on downstream tasks and Length-Adaptive Transformers that are further trained on downstream tasks with token drops. We focus on comparing the inference efficiency with three different metrics: the number of trainable parameters, CPU wall clock time, and the number of floating operations (FLOPs), which is independent of hardware and validated to have linear correlations with CPU latency on (Kim and Cho, 2021). We measured the average CPU latency across the validation dataset with Intel i9-9980XE using 18 threads.

	SQuAD v1.1	
Model	F1	FLOPs
BERT	88.39	1.00x
BERT$_{LAT}$†	89.12	0.86x
BERT$_{LAT}$*	87.39	0.41s
BERT$_{THAT}$†	89.23	0.68x
BERT$_{THAT}$*	87.53	0.34x
DistilBERT	85.69	1.00x
DistilBERT$_{LAT}$†	85.46	0.80x
DistilBERT$_{LAT}$*	84.69	0.55x
DistilBERT$_{THA}$†	85.87	0.92x
DistilBERT$_{THA}$*	84.70	0.50x
MobileBERT	89.31	1.00x
MobileBERT$_{LAT}$†	89.63	0.93x
MobileBERT$_{LAT}$*	88.69	0.56x
MobileBERT$_{THA}$†	89.99	0.84x
MobileBERT$_{THA}$*	89.13	0.51x

Table 1: Comparison of Length Adaptive Transformers and Token and Head Adaptive Transformers on SQuAD v1.1. $\dagger$ denotes the most efficient model while having the highest accuracy among the Pareto frontier. $\star$ denotes the most efficient model with accuracy within 1 percent of the standard model accuracy.

4.3 Hyperparameters

We fine-tune pre-trained BERT models for downstream tasks for 3 epochs without any pruning according to (Goyal et al., 2020). Then we further fine-tune BERT and DistilBERT on our token and head adaptive framework with sequence drop rate and layer drop rate set to 0.2, which is the same rate used for training a Length Adaptive Transformer. For MobileBERT, we train with a sequence drop rate set to 0.05 and a layer drop rate set to 0.15 to avoid an excessive amount of tokens pruned across 24 encoder layers. The head drop rate is set to a constant rate based on the number of layers of the models to have a single head remaining in the last layer. We fine-tuned with 2 randomly sampled intermediate pruning configurations in addition to upper and lower bound configurations to apply the sandwich training technique. We fine-tuned for 5 epochs on all the benchmarks. We use the batch size of 16 for BERT and 32 for DistilBERT and MobileBERT. The learning rate is set to 5e-5 for SQuAD and 2e-5 for GLUE benchmarks. For MobileBERT, the learning rate is set to 5e-5 for both SQuAD and GLUE benchmarks. The maximum sequence length is set to 384 for SQuAD v1.1 and 128 for SST-2, QQP, MNLI-mm, STS-B, and QNLI, and 64 for CoLA.

For evolutionary search, we run 30 iterations of evolutionary search with 30 mutation pruning configurations with a mutation probability of 0.5 and 30 crossover pruning configurations on each iteration to find the accuracy-efficiency Pareto frontier.

5 Results and Analysis

In this section, we evaluate the experiments and analyze the pruning results in terms of the accuracy-efficiency trade-offs and head distribution.

5.1 Pareto frontier

We compare the accuracy-efficiency Pareto frontier of Token and Head Adaptive Transformers and Length Adaptive Transformers on the SQuAD and GLUE benchmarks. As shown in Fig. 2, the Pareto curve of the Token and Head Adaptive Transformers have a larger area-under-curve (AUC) compared to those of the Length Adaptive Transformers. The Pareto frontier for the SQuAD benchmark shows that all the BERT models fine-tuned with the proposed framework generate joint pruning schemes with superior accuracy-latency trade-offs. It is noticeable in the MRPC Pareto curve that the sparse models with certain pruning configurations have even higher accuracy with significantly less number of floating point operations compared to the full model (rightmost point on Pareto curve) for both Length Adaptive Transformers and Token and Head Adaptive Transformers. Furthermore, the full model fine-tuned with the proposed framework achieves higher accuracy compared to those of Length Adaptive Transformers under the same fine-tuning setup with equal number of training iterations and learning rate. This shows that the models trained with the proposed framework generalize better compared to the standard fine-tuned models and the Length Adaptive Transformers. As we try to reduce the computational effort and reduce the GFLOPs, our models trained with the joint pruning policy remove unnecessary distractors both in terms of tokens and heads leading to boosting the accuracy. Beyond a certain point of pruning drops essential information as well showing the trend towards lowering accuracy consequently. This describes a general behavior that depends on the actual pruning that happens, and it is very data sensitive. This trend is also shown for the STS-B Pareto curve for BERT and DistilBERT.

Additionally, we find that BERT model fine-tuned with the joint pruning framework generates significantly better accuracy-efficiency trade-offs compared to the standard DistilBERT model when enough computational budget is allowed. For example, with the computational budget of 25 GFLOPs, BERT model outperforms DistilBERT by achieving 5 percent higher accuracy while requiring the same number of GFLOPs. Similar trends can be found in the other two benchmarks as well.

5.2 Maximizing efficiency gain

We also measure how much efficiency could be gained with less than 1 percent accuracy loss from the standard fine-tuned models. Table. 1 shows the accuracy and the efficiency gain of the standard Transformers, Length Adaptive Transformers, and Token and Head Adaptive Transformers on the SQuAD benchmark. To validate our proposed framework could find joint pruning configurations that maximize both accuracy and efficiency, we searched two pruning configurations for each model that maximize accuracy and efficiency gain respectively. † denotes the most efficient model while having the highest accuracy. $\star$ denotes the model with the highest efficiency gain that has an accuracy within 1 percent of the standard model accuracy. For all cases, Token and Head Adaptive Transformers achieves higher accuracy and higher efficiency gain. Fig. 2 provides further insights and shows the Pareto frontier curves of our models are shifted upper-left compared to the curves from Length-Adaptive Transformers. This shows that

Model	Accuracy/FLOPs							
	CoLA	STS-B	MRPC	SST-2	QNLI	MNLI-mm	QQP	GLUE
BERT	55.3/1.00x	87.3/1.00x	89.3/1.00x	92.6/1.00x	90.8/1.00x	83.0/1.00x	88.4/1.00x	83.8/1.00x
BERT$_{LAT}$†	57.7/0.38x	88.4/0.55x	89.0/0.39x	92.7/0.36x	91.1/0.55x	84.1/0.53x	89.6/0.60x	84.6/0.48x
BERT$_{LAT}$*	56.2/0.34x	86.6/0.36x	89.0/0.39x	92.4/0.35x	89.8/0.35x	83.2/0.34x	89.4/0.35x	83.8/0.35x
BERT$_{THAT}$†	58.1/0.31x	88.6/0.49x	89.0/0.32x	92.5/0.31x	91.1/0.46x	83.9/0.48x	89.5/0.41x	84.7/0.40x
BERT$_{THAT}$*	56.8/0.30x	86.9/0.30x	89.0/0.32x	91.6/0.30x	89.8/0.31x	83.3/0.31x	89.4/0.30x	83.8/0.31x
DistilBERT	40.2/1.00x	83.4/1.00x	83.2/1.00x	90.2/1.00x	87.1/1.00x	80.0/1.00x	87.1/1.00x	78.7/1.00x
DistilBERT$_{LAT}$†	44.1/0.54x	84.6/0.55x	82.4/0.54x	90.7/0.54x	88.1/0.63x	81.2/0.56x	88.6/0.59x	80.0/0.56x
DistilBERT$_{LAT}$*	44.1/0.54x	84.5/0.54x	82.4/0.54x	90.7/0.54x	87.8/0.54x	81.2/0.56x	88.5/0.54x	79.9/0.54x
DistilBERT$_{THAT}$†	45.9/0.46x	84.9/0.48x	83.3/0.52x	91.2/0.46x	88.0/0.59x	81.2/0.46x	88.6/0.46x	80.4/0.49x
DistilBERT$_{THAT}$*	45.9/0.46x	84.0/0.45x	82.4/0.45x	91.2/0.46x	86.6/0.45x	81.2/0.46x	88.6/0.46x	80.0/0.45x
MobileBERT	46.0/1.00x	84.0/1.00x	85.1/1.00x	91.3/1.00x	87.0/1.00x	81.1/1.00x	86.0/1.00x	80.1/1.00x
MobileBERT$_{LAT}$†	49.6/0.20x	86.9/0.39x	83.1/0.22x	92.4/0.25x	89.6/0.46x	83.0/0.71x	88.3/0.39x	81.6/0.37x
MobileBERT$_{LAT}$*	49.6/0.20x	83.1/0.20x	-	90.9/0.20x	86.2/0.23x	80.3/0.26x	85.7/0.20x	79.3/0.21x
MobileBERT$_{THAT}$†	53.6/0.25x	87.9/0.92x	85.6/0.33x	92.3/0.80x	90.2/0.65x	83.2/0.94x	88.3/0.35x	83.0/0.60x
MobileBERT$_{THAT}$*	50.1/0.21x	83.3/0.20x	84.3/0.22x	91.9/0.19x	86.1/0.22x	80.2/0.25x	86.3/0.19x	80.3/0.21x

Table 2: Results are evaluated on the test set of GLUE benchmark. $\dagger$ and $\star$ denote the same optimal models as described in Table 1.

our model trained with the joint pruning scheme produce models with less computational cost and higher accuracy. Our token and head adaptive BERT model achieves higher accuracy and only requires 34% of the number of floating point operations from the baseline model, which is 7% smaller than Length-Adaptive Transformer.

Table. 3 also shows the number of learnable parameters and CPU latency of the models with maximum efficiency gains. Our adaptive BERT model achieves higher accuracy while it is 15% faster and 20% smaller. Our adaptive DistilBERT and MobileBERT are 12% and 9% faster than Length-Adaptive Transformer models while achieving better accuracy.

The overall GLUE scores for the two pruning configurations with the highest accuracy and the maximum latency gain also validate the token and head adaptive model with joint pruning configurations achieves better efficiency gain and higher accuracy. MobileBERT trained with our joint token and head pruning scheme achieves a GLUE score of 83.0, which is 1.9 higher than the standard MobileBERT and 1.4 higher than Length-Adaptive Transformers.

5.3 Head distribution

We further analyze the joint pruning configurations and compare with length pruning schemes from Length-Adaptive Transformers. Fig. 3 shows the best pruning configurations of Length Adaptive and Token and Head Adaptive Transformers that maximize efficiency within 1 percent of the standard model accuracy. For SST-2 benchmark, the 3 token and head adaptive BERT-based models only

Model	Param(MB)	Lat(ms)	heads	tokens
BERT$_{LAT}$*	415.4	385.7	144	1846
BERT$_{THAT}$*	333.3	333.4	77	1782
DistilBERT$_{LAT}$*	253.2	251.4	72	1194
DistilBERT$_{THAT}$*	240.3	223.1	55	1174
MobileBERT$_{LAT}$*	93.8	360.4	96	5266
MobileBERT$_{THAT}$*	91.1	331.0	55	4891

Table 3: The number of parameters, CPU latency, the number of attention heads, and the number of tokens for BERT-based models that maximize efficiency on SQuAD benchmark.

retain a single head in the last layer. Table. 3 shows that 48%, 24%, and 43% of the attention heads are pruned from BERT, DistilBERT and MobileBERT respectively and still performs superior to the standard fine-tuned models and Length-Adaptive Transformers. On SQuAD benchmark, the token and head adaptive models progressively reduce the attention heads where more than 50% of the attention heads are pruned in the last layer.

The token and head adaptive MobileBERT tends to retain slightly more tokens in the center layers compared to Length Adaptive Transformer and reduce more tokens in the later layers. Our adaptive model have higher token drop rate. This shows the final models trained with our framework are robust to prune even more tokens along with attention heads.

6 Related Work

Transformers and BERT have outperformed existing language based models and achieved state-of-the-art results but at the cost of high computational power requirements. In order to make these models efficient in terms of memory and power footprint,

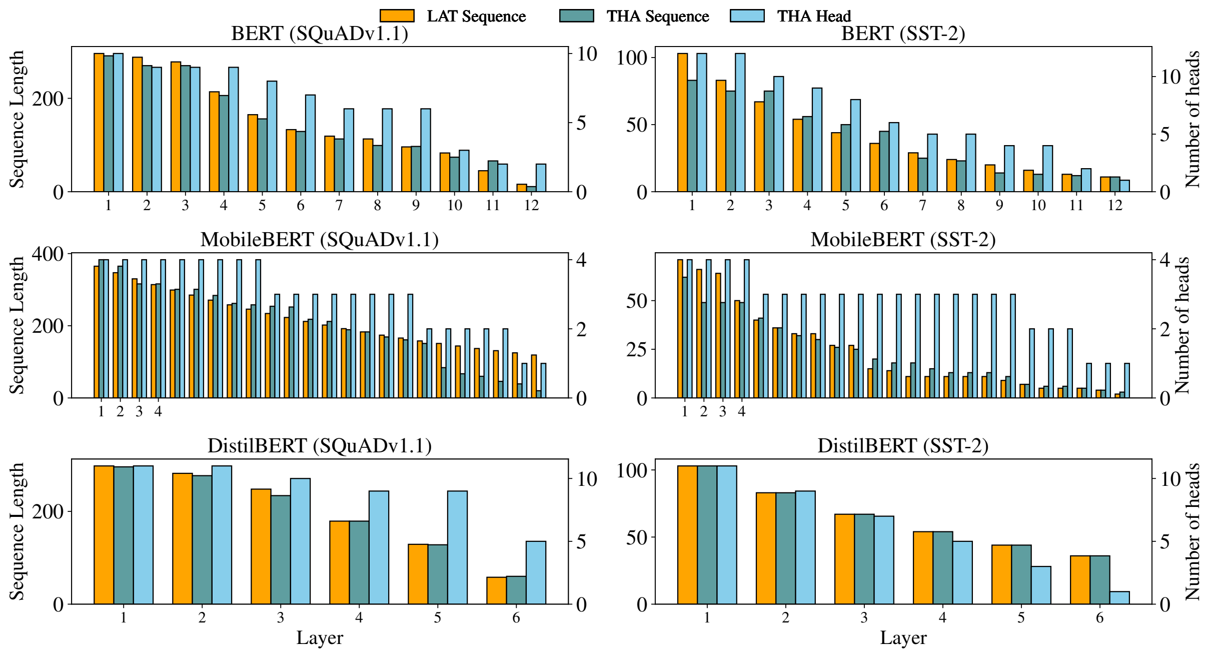

Figure 3: Comparison of the Sequence length configurations of the ⋆ models with the maximum efficiency gains on SQuAD and SST-2 benchmarks. LAT Sequence represents the token length configuration from Length-Adaptive Transformer while THA for Token and Head Adaptive Transformers. THA Head shows the retaining head distribution of Token and Head Adaptive models.

several strategies have been proposed. Here we discuss a few key studies.

While large networks are essential for the best performance, the model size becomes an obstacle when it comes to scaling and also leads to worse training time. ALBERT (Lan et al., 2020) targets this limitation and proposed parameter reduction training by factoring the embedding matrices and sharing the parameters across layers which resulted in a smoother up-scaling up of the base model as well as significantly less training time. However, in terms of inference latency, there is not much improvement. PoWER-BERT (Goyal et al., 2020) points out this fact and proposes an orthogonal approach of progressive fine-grained word-vector elimination that can reduce the inference time up to 4.5x without harming the accuracy much. However, PoWER-BERT requires repetitive training for each different constraint and is limited to sequence-level classification only. Length adaptive transformer or LAT (Kim and Cho, 2021) alleviates these shortcomings by introducing a LengthDrop method that automatically derives multiple sub-models using evolutionary search without requiring any re-training. They also proposed a Drop-and-Restore process to set aside the word-vectors so that they can be restored at the final layers extending the PoWER-BERT beyond classification tasks to a broader range of NLP tasks.

Parallel to these studies that deal with token pruning, another line of work investigates whether we need so many attention heads. (Voita et al., 2019) identifies the most important attention heads using layer-wise relevance propagation or LRP (Ding et al., 2017) and gradually prunes the head accordingly. In a complementary approach, (Michel et al., 2019) finds the most relevant heads by masking or ablating one or more heads and looking at their impact on performance. In this way, they calculate a proxy importance score and iteratively prune the heads according to that without degrading the performance significantly. Another group of studies concentrates on model architecture and parameter size. Where DistilBERT (Sanh et al., 2019) uses a teacher-student setting to transfer knowledge from a large model to its small student version to reduce the model parameters, MobileBERT (Sun et al., 2020) reduces it by leveraging the bottleneck structure.

While both token and head pruning improves the model's efficiency i.e. less inference time, lower memory, and power footprint, to the best of our knowledge, no study has tried to optimize both of them simultaneously which precisely lays down the basis of our work.

7 Conclusion and Future Work

In this work, we propose a joint token and head pruning framework to train robust Transformer models adaptive to token and head dropouts. Our idea is to calculate head importance from the self-attention operation and eliminate redundant heads from the retained tokens to further reduce the computational cost and train the model adaptive and perform better generalization. The final models trained with the proposed framework show better generalization and greater efficiency gains. Token and Head Adaptive Transformers are robust to prune out more tokens than Length-Adaptive Transformers along with the attention heads. This leads to superior accuracy-latency trade-offs and larger area-under-curve of the Pareto frontier found with an evolutionary search. We would also like to explore further in the direction of reducing the computational cost of large-scale models by eliminating redundant operations and applying efficient optimization techniques to search for efficient models with minimum resources.

Acknowledgements

This work was supported in part by the Center for Brain-inspired Computing (C-BRIC) and the Center for Research in Intelligent Storage and Processing in Memory (CRISP), two of the six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

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

Yanzhuo Ding, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1150–1159, Vancouver, Canada. Association for Computational Linguistics.

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

Saurabh Goyal, Anamitra Roy Choudhury, Saurabh M. Raje, Venkatesan T. Chakaravarthy, Yogish Sabharwal, and Ashish Verma. 2020. Power-bert: Accelerating bert inference via progressive word-vector elimination. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Katyanna Quach. 2020. Ai me to the moon... carbon footprint for 'training gpt-3' same as driving to our natural satellite and back. [online]. Available: https://www.theregister.com/2020/11/04/gpt3_carbon_footprint_estimate/. [Accessed: May-17-2022].

Gyuwan Kim and Kyunghyun Cho. 2021. Length-adaptive transformer: Train once with length drop, use anytime with search. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6501–6511, Online. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2016. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing @ NeurIPS 2019*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2158–2170, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

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. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

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*.

Jiahui Yu and Thomas S Huang. 2019. Universally slimmable networks and improved training techniques. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1803–1811.