Pruning Redundant Mappings in Transformer Models via Spectral-Normalized Identity Prior

Zi Lin[*]
Google Research
lzi@google.com

Jeremiah Zhe Liu[†]
Google Research & Harvard University
jereliu@google.com

Zi Yang
Google Research
ziy@google.com

Nan Hua
Google Research
nhua@google.com

Dan Roth
University of Pennsylvania
danroth@seas.upenn.edu

Abstract

Traditional (unstructured) pruning methods for a Transformer model focus on regularizing the individual weights by penalizing them toward zero. In this work, we explore *spectral-normalized identity priors* (SNIP), a structured pruning approach that penalizes an entire residual module in a Transformer model toward an identity mapping. Our method identifies and discards unimportant non-linear mappings in the residual connections by applying a thresholding operator on the function norm. It is applicable to any structured module, including a single attention head, an entire attention block, or a feed-forward subnetwork. Furthermore, we introduce *spectral normalization* to stabilize the distribution of the post-activation values of the Transformer layers, further improving the pruning effectiveness of the proposed methodology. We conduct experiments with BERT on 5 GLUE benchmark tasks to demonstrate that SNIP achieves effective pruning results while maintaining comparable performance. Specifically, we improve the performance over the state-of-the-art by 0.5 to 1.0% on average at 50% compression ratio.

1 Introduction

Natural Language Processing (NLP) has recently achieved great success by using the Transformer-based pre-trained models (Radford et al., 2019; Devlin et al., 2018; Yang et al., 2019; Clark et al., 2020). However, these models often consume considerable storage, memory bandwidth, and computational resource. To reduce the model size and increase the inference throughput, compression techniques such as knowledge distillation (Sanh et al., 2019; Sun et al., 2019; Tang et al., 2019; Jiao et al., 2019; Sun et al., 2020) and weight pruning (Guo
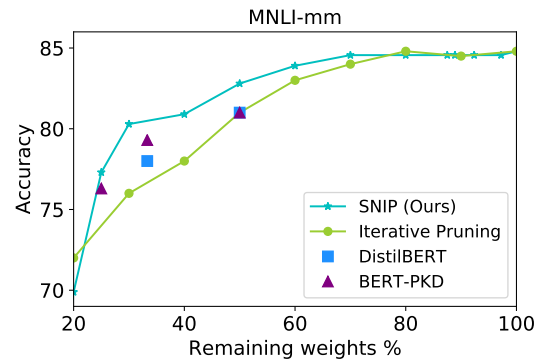

Figure 1: Comparison of selected knowledge distillation methods (DistilBERT (Sanh et al., 2019) and BERT-PKD (Sun et al., 2019)) and iterative pruning methods (Iterative Pruning (Guo et al., 2019) and our proposed method) in terms of accuracy at various compression rate using MNLI test set. knowledge distillation methods require re-distillation from the teacher to get each single data point, whereas iterative pruning methods can produce continuous curves at once.

et al., 2019; Wang et al., 2019; Gordon et al., 2020; Sanh et al., 2020) have recently been developed.

Knowledge distillation methods require the specification of a student network with a smaller architecture, which often has to be identified by a tedious sequence process of trial-and-error based decisions. By comparison, the iterative pruning methods gradually prune the redundant model weights or layers from the full-size model, and provide a full picture of the trade-off between the task performance and the model size with a single training process, as illustrated in Figure 1. This allows the iterative pruning methods to easily determine the most compact architecture given a required level of model performance.

However, many of the existing pruning methods rely on classic regularizers that act on the individual weights by penalizing them to zero (Guo et al., 2019; Sanh et al., 2020). As a result, the pruned

---

[*]Work done as part of the Google AI Residency.
[†]Work done at Google Research.

model tends to maintain the same architecture as the original model despite the reduced parameter count, which does not practically lead to an improvement in inference latency (Wen et al., 2016).

This leads to a question: is it possible to perform more structured pruning on a Transformer model to modify its model architecture (e.g., reducing width and depth)?

To this end, we notice that many previous works have suggested that the learned Transformer models often have much redundancy (Tenney et al., 2019; Liu et al., 2019a; Jawahar et al., 2019; Kovaleva et al., 2019). For example, Michel et al. (2019); Voita et al. (2019) found that most of the attention heads in a Transformer model can be removed without significantly impacting accuracy, and Tenney et al. (2019) found that while the earlier layers and the later layers in a BERT model play clear roles in extracting either low-level or task-specific linguistic knowledge, the roles of the intermediate layers are less important. These observations have motivated the idea that Transformer models may exhibit considerable *structural redundancy* - i.e., some layers can be removed during the training without harming the final performance.

In this paper, we propose a *structured* pruning method that reduces the architecture of a Transformer by directly targeting its sub-modules as a whole - for example, a single attention head, an attention module, a feed-forward subnetwork, etc. We take an approach we call *spectral-normalized identity prior* (SNIP), which imposes a function-level prior centered around the identity function on the aforementioned modules.

Specifically, we take the advantage of the residual blocks ($\mathcal{F}(x) + x$) within a Transformer layer and compress them to *strict identity mappings* (Yu et al., 2018) by identifying the residual blocks whose nonlinear mapping's absolute values ($|\mathcal{F}(x)|$) mostly fall below a threshold ϵ. With this strategy, the weights of the Transformer model can still be under-regularized when using simple L_1 or L_2 based regularizers, which cause the distribution of the post-activation values prone to be noisy even after layer normalization (Ba et al., 2016). To address this issue, we further leverage *spectral normalization* (Miyato et al., 2018) to stabilize the distribution of the post-activation values by regularizing the largest singular value of the weight matrices.

We use BERT (Devlin et al., 2018) as a case study in this paper. Across multiple tasks in the GLUE benchmark (Wang et al., 2018), SNIP improves the performance over the state-of-the-art by 0.5 to 1.0% on average at 50% compression ratio. [1] We also show that spectral normalization results in more sparse and regulated layer mappings during pre-training. We compare the remaining model components across the tasks at a fixed compression ratio in an ablation study, and show that the remaining components are similar but not identical.

Our contributions are three-fold: First, we introduce *identity-inducing prior*, a structured pruning approach that imposes identity-inducing regularization on the Transformer mappings as a whole rather than its individual weights. Second, we show that through a novel combination with the *spectral normalization* regularization, the resulting *spectral-normalized identity prior* (SNIP) leads to well-regularized weight distribution and sparse layer mappings in a BERT model. Finally, we conduct thorough experiments to validate the SNIP approach over 5 standard NLU tasks. Our results suggest that different model components in a Transformer play critically different roles across tasks, suggesting the importance of performing task-specific pruning to obtain an architecture that is the most suitable for the target task.

2   Related Work

Pre-trained Language Model Compression The major existing efforts to compress pre-trained language models such as BERT include knowledge distillation (Ba and Caruana, 2014; Hinton et al., 2015) and pruning (Iandola et al., 2016; Veit and Belongie, 2017).

The knowledge distillation approach enables the transfer of knowledge from a large *teacher* model to a smaller *student* model. Such attempts have been made to distill BERT models, e.g., DistilBERT (Sanh et al., 2019), BERT-PKD (Sun et al., 2019), Distilled BiLSTM (Tang et al., 2019), TinyBERT (Jiao et al., 2019), MobileBERT (Sun et al., 2020), etc. All of these methods require carefully designing the student architecture. Furthermore, to choose which intermediate results that the student model can learn from, e.g., the outputs of each layer, the attention maps, is still under discussion.

Similar to other pruning-based methods, our

---

[1]Open-source code can be found at `https://github.com/google-research/google-research/tree/master/snip`

method can iteratively remove the least important weights or connections, explore the full spectrum of trade-offs, and find the best affordable architecture in one shot.

Many language representation model pruning methods focus on individual components of the weight matrices. For example, Guo et al. (2019) integrates reweighted L_1 minimization with a proximal algorithm to search sparsity patterns in the model; Gordon et al. (2020) uses magnitude weight pruning, which compresses the model by removing weights close to 0; Sanh et al. (2020) applies deterministic first-order weight pruning method where both weights with low and high values can be pruned. A very few works try structured weight pruning, e.g., Wang et al. (2019) proposes a structured pruning approach based on low-rank factorization and augmented Lagrangian L_0 norm regularization. On the other hand, there also exist works that prune a coherent set of sub-modules in the Transformer model. For example, Michel et al. (2019) and Voita et al. (2019) propose to prune individual attention heads either manually via head importance score, or automatically via a relaxed L_0 regularization. Fan et al. (2020) applies random pruning to the entire layers. In contrast, our method allows finer-grained structured pruning on Transformer modules (i.e., both attention heads and feed-forward layers) and propose to improve the mathematical property of a Transformer (i.e., Lipschitz condition) for more effective pruning.

Other compression approaches include weight sharing (Liu et al., 2019b), quantization (Zafrir et al., 2019; Shen et al., 2019) and neural architecture search (Chen et al., 2020), but are not within the discussion of this paper. We refer interested readers to Ganesh et al. (2020) for further details.

Applications of Spectral Normalization Spectral normalization is first proposed for generative adversarial network (GAN) as a regularization technique to stabilize the discriminator training (Miyato et al., 2018). It was later applied to improve the performance of the other types of generative neural networks (Zhao et al., 2018; Behrmann et al., 2019), and was analyzed theoretically in the context of adversarial robustness and generalization (Farnia et al., 2018).

Spectral normalization regularizes the Lipschitz condition of the model mappings and is known to benefit model generalization under both the classic and the adversarial settings (Sokolić et al., 2017;

Cisse et al., 2017; Oberman and Calder, 2018; Neyshabur et al., 2017). In this paper, we will explore the benefit of spectral regularization for improving the effectiveness of pruning.

3 Methods

In this section, we first briefly review the basic Transformer layers in Vaswani et al. (2017) (3.1). We then introduce our identity prior into Transformer's residual connections using ϵ threshold (3.2). In section 3.3, we give mathematical foundations to the spectral normalization and show how it could help with our identity prior. Finally, to put it all together, we establish our structured iterative pruning methods for BERT fine-tuning (3.4).

3.1 Background: Transformer Layer

Transformer-based models are usually comprised of a stack of Transformer layers. A Transformer layer takes on a sequence of vectors as input, first passes it through a (multi-head) self-attention sub-layer, followed by a position-wise feed-forward network sub-layer.

Self-attention sub-layer The attention mechanism can be formulated as querying a dictionary with key-value pairs. Formally,

$$\text{Att}(Q, K, V) = \text{softmax}(QK^T/\sqrt{d_H}) \cdot V$$

where d_H is the dimension of the hidden representations. Q, K, and V represent query, key, and value. The multi-head variant of attention (MHA) allows the model to jointly attend to information from different representation sub-spaces, defined as

$$\text{MHA}(Q, K, V) = [\text{head}_1, \dots, \text{head}_A]W^O$$
$$\text{head}_k = \text{Att}(QW_k^Q, KW_k^K, VW_k^V)$$

where $[\cdot, \cdot]$ is the concatenation operator, $W_k^Q \in \mathbb{R}^{d_H \times d_K}$, $W_k^K \in \mathbb{R}^{d_H \times d_K}$, $W_k^V \in \mathbb{R}^{d_H \times d_V}$, and $W^O \in \mathbb{R}^{Hd_V \times d_H}$ are projection parameter matrices, A is the number of heads, and d_K and d_V are the dimensions of key and value.

Position-wise FFN sub-layer In addition to the self-attention sub-layer, each Transformer layer also contains a fully connected feed-forward network, which is applied to each position separately and identically. This feed-forward network consists of two linear transformations with an activation function σ in between. Specially, given vectors

$\mathbf{x}_1, \ldots, \mathbf{x}_n$, a position-wise FFN sub-layer transforms each $\mathbf{x}_i$ as $\mathrm{FFN}(\mathbf{x}_i) = \sigma(\mathbf{x}_i W_1 + b_1) W_2 + b_2$, where W_1, W_2, b_1 and b_2 are parameters.

We should also emphasize that a residual connection (He et al., 2016b) and a layer normalization (Ba et al., 2016) are applied to the output of both MHA or FNN sub-layers. The residual connection plays a key role in learning strict identity mapping (detailed in Section 3.2), while layer normalization and spectral normalization (detailed in Section 3.3) together ensure regulated magnitude of activation outputs for improved pruning stability.

3.2 Identity-inducing Prior for Transformer

The design of residual connection can provide us with a promising way to find identity mappings for the Transformer model. Specifically, residual connection (He et al., 2016b) can be formalized as $\mathcal{H}(\mathbf{x}) = \mathcal{F}(\mathbf{x}) + \mathbf{x}$, where $\mathcal{F}$ could be either MHA or FFN and $\mathcal{H}$ is the sub-layer output. As illustrated in He et al. (2016a), if an identity mapping is optimal, it is easier to push the residual to zero than to fit an identity mapping by a stack of traditional non-linear layers.

We leverage ϵ-ResNet (Yu et al., 2018), a strict identity mapping mechanism that sparsifies the layer output by inducing a specific threshold ϵ as the *identity prior*. Specifically, we turn the residual connection $\mathcal{H}(\mathbf{x})$ into $\mathcal{S}_\epsilon(\mathcal{F}(\mathbf{x})) + \mathbf{x}$, where

$$\mathcal{S}_\epsilon(\mathbf{v}) = \begin{cases} 0 & \text{if } |\mathbf{v}_i| < \epsilon, \forall i \in 1, ..., |\mathbf{v}|, \\ \mathbf{v} & \text{otherwise,} \end{cases}$$

where $\mathbf{v}_i$ is the i-th element of the vector $\mathbf{v}$. Here a sparsity-promoting function $\mathcal{S}$ is applied to dynamically discard the non-linearity term based on the activations. When all the responses in the non-linear mapping $\mathcal{F}(\mathbf{x})$ is below a threshold ϵ, then $\mathcal{S}(\mathcal{F}(\mathbf{x})) = 0$, otherwise, the original mapping $\mathcal{S}(\mathcal{F}(\mathbf{x})) = \mathcal{F}(\mathbf{x})$ was used as the standard residual network.

To implement $\mathcal{S}$, we put an extra binary gate layer t_ϵ upon $\mathcal{F}(\mathbf{x})$ by stacking additional rectified linear units (ReLU), following Srivastava et al. (2015). In particular,

$$t_\epsilon(\mathbf{v}) = 1 - \mathrm{ReLU}(1 - L \max_{1 \le i \le |\mathbf{v}|} \mathrm{ReLU}(|\mathbf{v}_i| - \epsilon))$$

where L refers to a very large positive constant (e.g., $1e5$ in our experiments). Then, $\mathcal{S}_\epsilon(\mathcal{F}(\mathbf{v}))$ has the following form:

$$\mathcal{S}_\epsilon(\mathcal{F}(\mathbf{x})) = t_\epsilon(\mathcal{F}(\mathbf{x})) \mathcal{F}(\mathbf{x})$$

Recall that each layer of Transformer consists of two residual blocks, namely, the self-attention sub-layer and the position-wise FFN sub-layer. We apply the ϵ network directly to the residual block in the FFN sub-layer, i.e.,

$$\mathcal{H}_{\mathrm{FFN}}(\mathbf{x}) = \mathcal{S}_{\epsilon_{\mathrm{FFN}}}(\mathrm{FFN}(\mathbf{x})) + \mathbf{x},$$

When applying it to the attention sub-layer, we place $\mathcal{S}$ to each single attention head, which allows us to prune any subset of attention heads, i.e.,

$$\mathcal{H}_{\mathrm{ATT}}(\mathbf{x}) = \sum_{i=1}^{A} \mathcal{S}_{\epsilon_{\mathrm{ATT}}}(\mathrm{head}_i W_i^O) + \mathbf{x}$$

Here, W_i^O is the output weight assign to the i-th attention head, i.e., $W^O = [W_I^O, W_2^O, ..., W_A^O]$. If $\mathcal{S}_{\epsilon_{\mathrm{ATT}}}(\mathrm{head}_i W_i^O) = 0$, this means that the i-th attention head does not contribute to the output of the attention layer and thus could be pruned out.

In our experiment, we set different values to ϵ_{ATT} and ϵ_{FFN}, since the absolute outputs of attention and FFN layers lay in different scalars, as illustrated in Figure 2.

3.3 Spectral Normalization

The sparsity-inducing function $\mathcal{S}_\epsilon(\mathcal{F}(\mathbf{x}))$ in the ϵ-ResNet has been found to work well for randomly initialized neural network, where the initial weight matrix of the non-linear mappings for all layers was distributed within a consistent range, and thus facilitates a natural separation between the function norms $|\mathcal{F}|$ of the important and unimportant non-linear mappings in the residual blocks during training (Yu et al., 2018). This is, however, not the case for the weight distribution of a pre-trainned model like BERT, where the weight distributions between different layers have already diverged during pre-training, which is likely due to the specialization of layer functionalities under the masked language modeling (MLM) training (Tenney et al., 2019).

Indeed, in our preliminary experiments, we observed that the proposed identity-inducing prior ϵ is not effective for a BERT model initialized from a classic pre-training checkpoint. As shown in Table 1, we found the distribution of the function norms $P(|F(\mathbf{x})_i|)$ for the attention layers to be densely clustered within a small range $((0, 2))$ and with no clear separation between the function norm for the important and unimportant non-linear residual mappings. On the other hand, the norm distributions
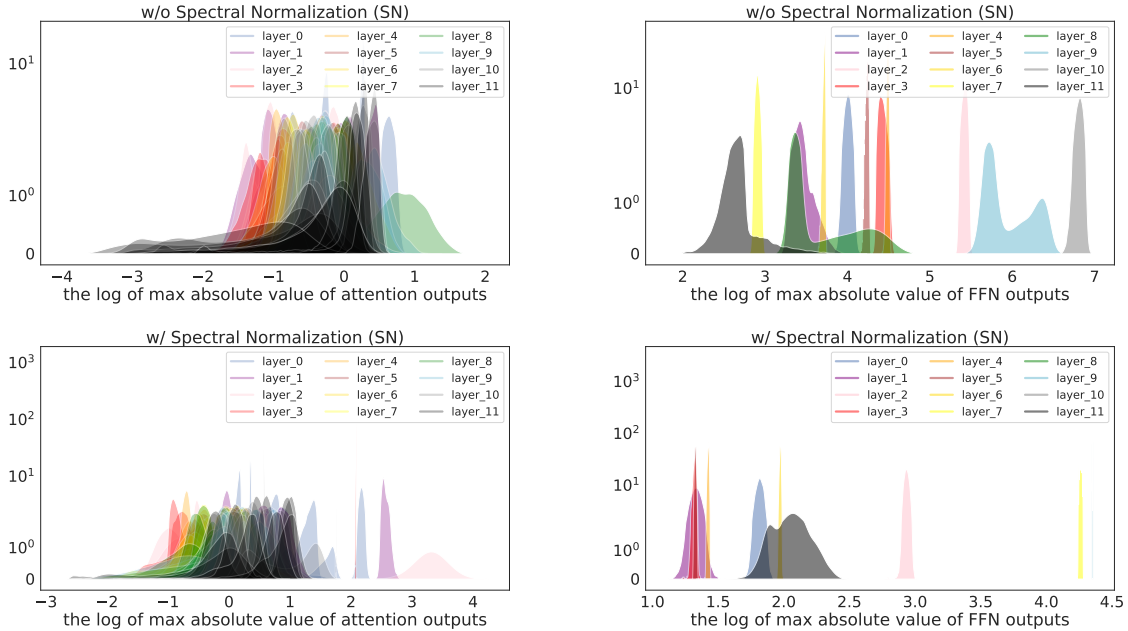
Figure 2: Distribution of the max absolute value of the attention/FFN outputs on the Stanford Sentiment Treebank (SST) training set, based on model $BERT_{BASE}$ (w/o SN) and spectral normalized $BERT_{BASE}$ (w/ SN). For attention layers, different heads are plotted separately but with the same color in the same layer for clarity. After SN, the norm distribution $P(|\mathcal{F}_k(\mathbf{x})|)$ shows a better separation (bottom v.s. top), and the norms distribution for the FFN layers are stabilized within a much smaller range (bottom right v.s. top right).

SN	MLM	SST-2	QQP	MRPC	QNLI	MNLI
w/o	71.93	92.7	90.6	90.9	91.6	84.7
w/	74.16	91.9	90.1	90.8	91.6	85.1

Table 1: Evaluate the performance of $BERT_{BASE}$ (w/o SN) and the spectral normalized $BERT_{BASE}$ (w/ SN, using $\lambda(W) = 5$), respectively masked language modeling accuracy on pre-training data and accuracy of fine-tuning on 5 natural language understanding tasks (details could be found in Section 4.1).

for different FFN layers were found to vary wildly, creating challenges for selecting a proper set of ϵ's in practice.

The above observations motivate us to identify an effective method to stabilize the norm distributions of BERT model layers. In this work, we consider *spectral normalization* (SN), an approach that directly controls the Lipschitz norm of a non-linear mapping $\mathcal{F}$ by regularizing the spectral behavior of its weight matrices (Miyato et al., 2018).

Specifically, for a weight matrix W, its spectral norm $\lambda(W)$ is defined as its largest singular value:

$$\lambda(W) = \max_{x \neq 0} \frac{||W\mathbf{x}||_2}{||\mathbf{x}||_2}.$$

We say a function $\mathcal{F}$ is L-Lipschitz if $|\mathcal{F}(\mathbf{x}_1) - \mathcal{F}(\mathbf{x}_2)|/||\mathbf{x}_1 - \mathbf{x}_2|| \leq L$, for all possible $(\mathbf{x}_1, \mathbf{x}_2)$

pairs from the feature space, and we call the smallest possible L the Lipschitz norm of $\mathcal{F}$, denoted as $|\mathcal{F}|_{Lip}$. Consequently, for a neural network mapping $\mathcal{F}(\mathbf{x}) = \sigma(W\mathbf{x} + b)$ with an contractive activation function σ, its Lipschitz norm is upper-bounded by $\lambda(W)$ (Miyato et al., 2018):

$$|\mathcal{F}|_{Lip} = |\sigma(W\mathbf{x}+b)|_{Lip} \leq |W\mathbf{x}+b|_{Lip} \leq \lambda(W).$$

For BERT models, since the layer input $\mathbf{x}$ follows a distribution of zero mean and unit variance due to layer normalization (Ba et al., 2016), a non-linear mapping's L_1 norm $|\mathcal{F}|$ is roughly proportional to its Lipschitz norm $|\mathcal{F}|_{Lip}$, which is controlled by $\lambda(W)$. Therefore, we are able to have a better control of the maximum of $|\mathcal{F}(\mathbf{x})|$ for identifying a good ϵ. Furthermore, the regularization is achieved by that the layer weights are simply divided by their corresponding spectral norm in SN, i.e., $\hat{W} = W/\hat{\lambda}(W)$, adding no additional trainable parameter to the original model.

We apply SN to both pre-training and fine-tuning of the BERT model, and on the weights in both the attention and the FFN layers. As shown in Table 1, compared to the original $BERT_{BASE}$ without SN, adding SN to a $BERT_{BASE}$ model has resulted in improved pre-training performance and competitive fine-tuning performance. To illustrate the effect
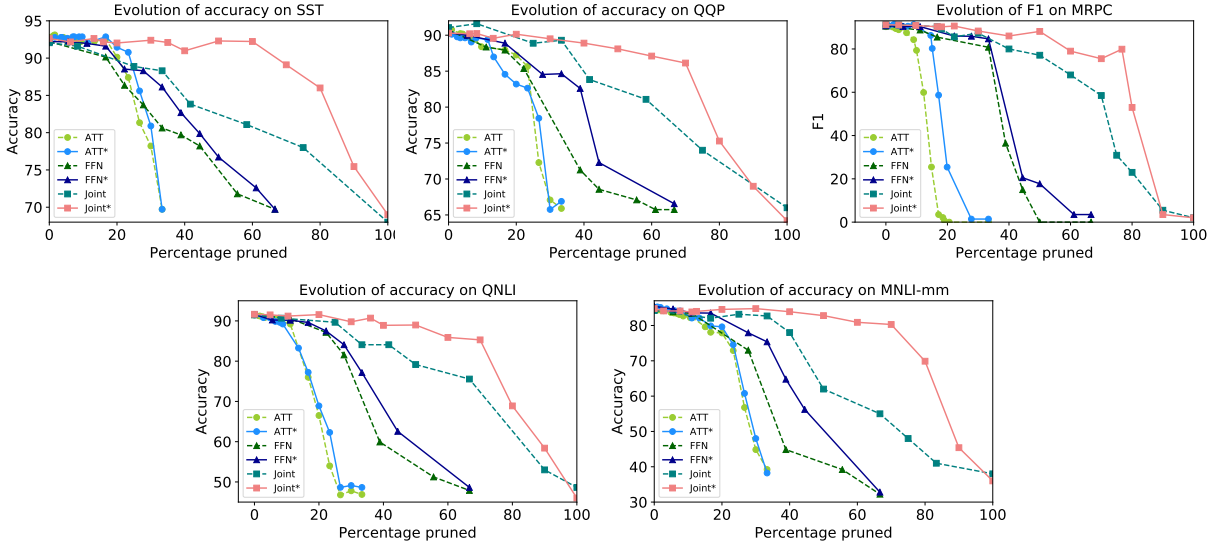
Figure 3: Evolution of performance when separately pruning attention heads (ATT) and FFN layers (FFN), and joint pruning ATT and FFN (Joint) on 5 GLUE dev sets. * means applying spectral normalization. The percentage pruned (x-axis) is calculated based on the parameters of attention and FFN sub-layers as the full set (excluding embedding, pooler and classifier layers). Joint pruning with spectral normalization (Joint*) could prune more parameters than the original BERT (Joint) to preserve the same accuracy.

of SN on the distribution of function norms of the BERT model, we plot $|\mathcal{F}(\mathbf{x})|$ for the training set of the Stanford Sentiment Treebank in Figure 2. As shown, after SN, the norm distribution $P(|\mathcal{F}_k(\mathbf{x})|)$ between the important and the unimportant nonlinear mappings shows a better separation (Figure 2, bottom), the norms distribution for the FFN layers are now stabilized within a much smaller range (Figure 2, bottom right).

3.4 Structured Iterative Pruning

We use a simple pruning method that greedily and iteratively prunes away attention heads and FFN layers to avoid impractical combinatorial search, where two dynamic estimations are conducted for ϵ and model architecture respectively. One iteration contains four substeps:

1. Estimate ϵ given current model architecture and training data. Specifically, We sort the attention heads and FFN layers by their mean activation outputs, and set ϵ to the k-th smallest mean activation. Larger k leads to more mappings being pruned in one iteration, which makes the retraining more difficult to recover the performance, but leads to fewer pruning iterations.

2. Train the model with identity-inducing prior by using the selected ϵ.

3. Estimate a smaller architecture given current ϵ and training data. Specifically, we estimate the module usage by counting the number of times each residual block has been learned to become a strict identity mapping across mini-batches in the training set. We prune residual blocks whose usage rate below a threshold θ. When a residual block produces a negligible response, the ϵ function will start producing 0 outputs. As a result, the weights in this block will stop contributing to the cross-entropy term. Consequently, the gradients will only be based on the regularization term and lead to weight collapse.

4. Retrain the model with the pruned residual blocks completely removed from the architecture. This is critical — if the pruned network is used without retraining, accuracy is significantly impacted. Also, during retraining, it is better to retain the weights from the initial training phase for the connections that survived pruning than it is to re-initialize them (Han et al., 2015).

4 Experiments

4.1 Experimental Settings

In the experiments, we apply the same architecture and the base settings from the original BERT$_{\text{BASE}}$

724

Model	FLOPS	SST-2$_{acc}$	QQP$_{acc}$	MRPC$_{F_1}$	QNLI$_{acc}$	MNLI-mm$_{acc}$
BERT$_{BASE}$	22.5B	-0.0%/92.7	-0.0%/90.6	-0.0%/90.9	-0.0%/91.6	-0.0%/84.7
TinyBERT$_6$ w/o DA (Jiao et al., 2019)[2]	1.2B	-	-	-50.0%/86.0	-	**-50.0%/84.4**
DistilBERT$_6$ (Sanh et al., 2019)	11.3B	-50.0%/91.3	-50.0%/88.5	-50.0%/87.5	-50.0%/89.2	-50.0%/82.2
BERT-PKD$_6$ (Sun et al., 2019)	11.3B	-50.0%/91.5	**-50.0%/88.9**	-50.0%/86.2	-50.0%/89.0	-50.0%/81.0
BERT-PKD$_3$ (Sun et al., 2019)	7.6B	-75.0%/87.5	-75.0%/87.8	-75.0%/80.7	-75.0%/84.7	-75.0%/76.3
FLOP (Wang et al., 2019)	15B	-35.0%/92.1	-	-35.0%/88.6	-35.0%/89.0	-
MvP (Sanh et al., 2020) †	N/A	-	-97.0%/89.2	-	-	-97.0%/79.7
SNIP (w/ SN)	13.2 - 14.5B	**-50.0%/91.8**	**-50.0%/88.9**	**-50.0%/88.1**	**-50.0%/89.5**	-50.0%/82.8
SNIP (w/ SN)	8.3 - 9.1B	-75.0%/88.4	-75.0%/87.8	-75.0%/81.2	-75.0%/84.6	-75.0%/78.3
SNIP (w/o SN)	16.8 - 18.2B	-30.0%/91.3	-38.7%/89.5	-39.7%/89.9	-26.1%/90.8	-32.3%/83.5
SNIP (w/ SN)	13.4 - 16.1B	-56.7%/91.7	-40.7%/89.7	-46.7%/89.9	-36.0%/90.7	-39.3%/83.9

Table 2: The compression results including model efficiency (percentage of reduced parameters) and performance from the GLUE dev results, and the MNLI result is evaluate for mismatched-accuracy (MNLI-mm). BERT$_{BASE}$ indicates the results of the fine-tuned BERT$_{BASE}$ in our implementation. The number of model parameters includes the attention and FFN sub-layers but excludes the embedding, pooler and classifier layers. The bold numbers indicate the best performance for keeping 50% of the parameters respectively. Rows 1–5 are knowledge distillation methods, and Rows 6 and 7 are pruning methods, where † means unstructured pruning [3].

(Devlin et al., 2018), and fine-tune each task independently. More details could be found in Appendix.

For pre-training, we use the same data as BERT, which consists of 3.3 Billion tokens from Wikipedia and BooksCorpus (Zhu et al., 2015). Similar to the standard BERT practice, we conduct the pre-training only once and from scratch (i.e., no second pre-training). We use dynamic token masking with the masked positions decided on-the-fly instead of during preprocessing. Also, we did not use the next sentence prediction objective proposed in the original BERT paper, as recent work has suggested it dost not improve the scores (Yang et al., 2019; Liu et al., 2019b).

For fine-tuning tasks, we focus on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018) in the main text since it is a thoroughly studied setting in many pruning/distillation work, thereby allowing comprehensive comparison. We conduct experiments on the subset of GLUE , classified into three categories:

1. Sentiment analysis: Stanford Sentiment Treebank (SST) (Socher et al., 2013);

2. Paraphrasing: Quora Question Pairs (QQP) (Chen et al., 2018) and Microsoft Research Paraphrase Corpus (MRPC) (Dolan and Brockett, 2005);

3. Natural language inference: Question Natural Language Inference (QNLI) (Chen et al., 2018), and Multi-genre Natural Language Inference (MNLI) (Williams et al., 2017).

The detailed description of downstream tasks could be found in Appendix. The reason for choosing this subset is that we found the variance in performance for those tasks lower than the other GLUE tasks.

We used the hyperparameters from Clark et al. (2020) for the most part. Since we run the training iteratively, we set train epoch as 1 for most of the task but 3 for task MRPC consider that the size of the datasets is much smaller than other tasks.

For ϵ and architecture estimations in Section 3.4, we set $k = 1$ and $\theta = 0.95$ in our experiment.

4.2 Results

We compare the pruning results on non-normalized and normalized BERT models on the 5 GLUE tasks, as shown in Figure 3, which includes separate pruning for attention heads and FFN sub-layers and joint pruning of both modules. The results demonstrate the advantage of spectral normalization.

To put it all together, in Table 2, we further show the simplest architecture we could get when allowed at most 1% in terms of performance degradation. This also means that further slight pruning will have a noticeable impact on the final results. We find that spectral normalization can lead to a better trade-off between parameter size and performance. Specifically, for the same ideal performance shown in the last two rows in Table 2, spec-

---

[2]TinyBERT (Jiao et al., 2019) utilizes data augmentation (DA), which makes it unfair to have a direct comparison. We only listed their results without DA.

[3]We do not list the results in Guo et al. (2019); Gordon et al. (2020) since they show the results in scatter plots instead of the exact numbers, and similar to MvP (Sanh et al., 2020), they are of unstructured pruning, which could not speed up the inference in practice.
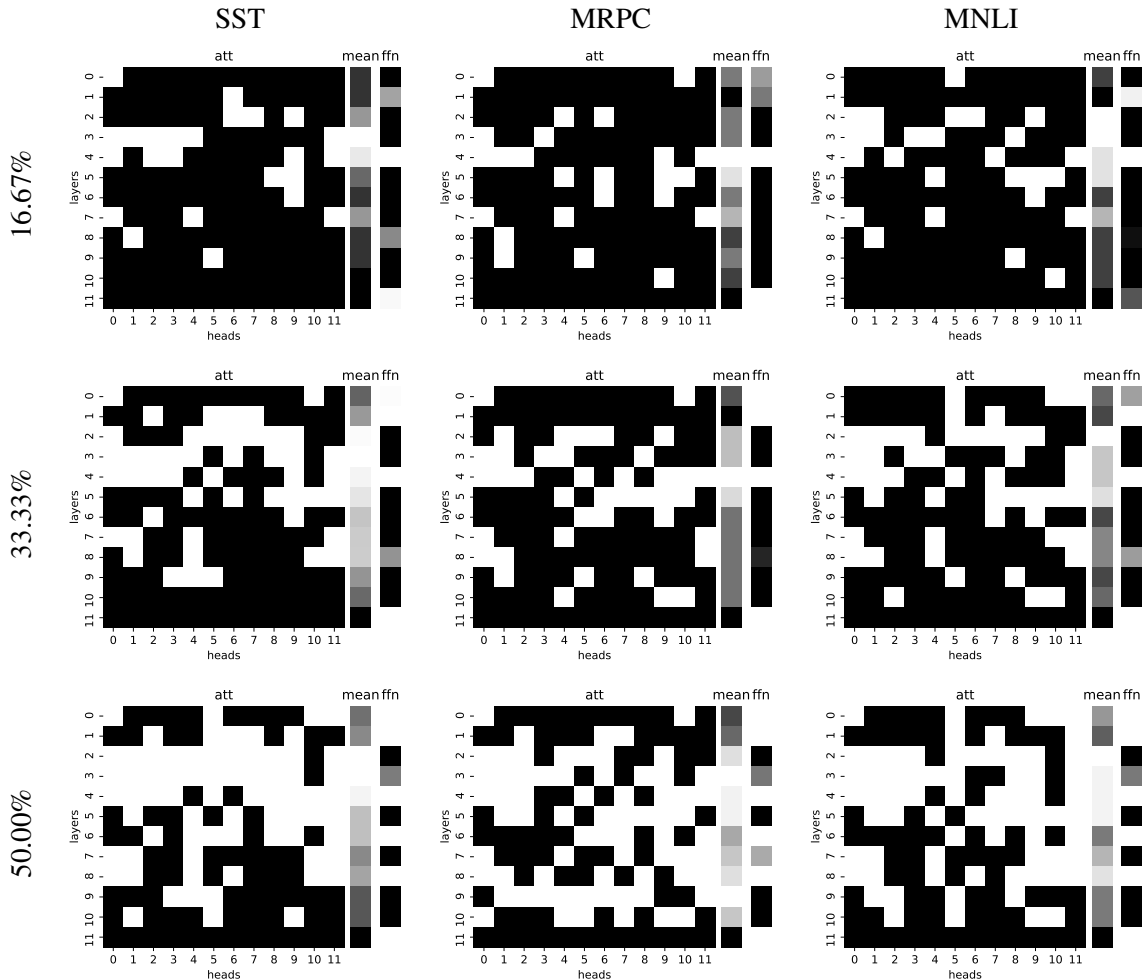
Figure 4: Pruning map of attention heads (`att`) and FFN layers (`ffn`) during pruning on task SST-2, MRPC and MNLI dev sets. `mean` is the mean of attention heads of each layers. The left column shows the percentage of pruned parameters. Color white indicates that this attention head or FFN layer has been entirely pruned. For different tasks, the distribution of important attention heads is similar, but not identical, and certain FFN sub-layers might be more amenable to compression, even if the corresponding self-attention sub-unit is not.

tral normalized BERT could on average be pruned 12% parameters more than the original BERT.

We also list other compression methods in the table for comparison while most of them are of knowledge distillation and have a pre-defined fixed size of the compressed model. Our methods provide the flexibility to choose the best architecture and take advantage of finding the inflection point during pruning, while compared with other pruning methods we could practically speed up the inference time since we use a structured pruning.

4.3 Analysis

In this section, we investigate the contribution of: (1) single attention head pruning, and (2) split pruning for attention heads and FFN sub-layers.

Single Head Attention Pruning Multi-head self-attention is a key component of Transformer, where each attention head potentially focuses on different parts of the inputs.

The analysis of multi-head attention and its importance is challenging. Previous analysis of multi-head attention considered the average of attention weights over all heads at given position or focused only on the maximum attention weights (Voita et al., 2018; Tang et al., 2018), or explicitly takes into account the varying importance of different heads (Voita et al., 2019). Michel et al. (2019) has proved that attention heads can be removed without significantly impacting performance, but they mainly focus on machine translation and NLI.

To understand whether and to what extent attention heads play consistent and interpretable roles when trained on different downstream tasks, we
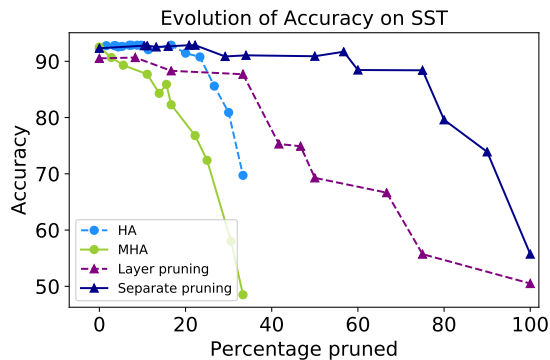
Figure 5: Evolution of accuracy on the dev set of SST-2 for (1) pruning the whole attention layers (HA); (2) pruning single attention heads (MHA); (3) pruning the whole layer (Layer pruning) and (4) pruning the attention and FFN sub-layers separately (Separate pruning).

pick one task from the sentiment analysis, paraphrasing and natural language inference respectively, plot the pruned attention heads during training and show the dynamic process in Figure 4. We can find that though for different tasks, the distributions of important attention heads are similar, as reflected in the mean of each attention layer, the exact pruned attention heads are actually different. This also indicates that splitting the pruning for attention heads could have more flexibility for the model to find an optimal architecture.

Separate Pruning for Attention and FFN Decoupling and then individually studying the self-attention and the FFN sub-layers is important for understanding the progressive improvement they provide. As can be observed from Figure 3, for most of the tasks, pruning FFN layers damages the performance more than the attention layers, indicating that the compression technique for the Transformer model tends to be more effective on the attention layers (Voita et al., 2018; Michel et al., 2019), than the FFN layers (Ganesh et al., 2020).

In Figure 4, similar to attention heads, we further plot the pruning map of FFN layers. We find that, certain FFN sub-layers might be more amenable to compression, even if the corresponding self-attention sub-unit is not. For example, in all the tasks, the FFN layers near the ends of input or output are more likely to be pruned, while this does not hold for the corresponding self-attention layers.

Finally, we compare between single head attention pruning and separate pruning for attention/FFN, and show the evolution of performance for single head attention pruning (HA), multi-head

attention pruning (MHA), separate attention/FFN pruning, and whole layer pruning respectively in Figure 5. We find that MHA and separate pruning perform much better than HA and layer pruning.

5 Conclusion

In this work, we propose a structured pruning method for compressing Transformer models, which prunes redundant mappings via spectral-normalized identity priors (SNIP). We achieve effective pruning results on BERT fine-tuning while maintaining comparable performance. Our work shows the importance of the mathematical properties of the Transformer model (specifically, the Lipschitz condition) on the effectiveness of pruning.

Additionally, we quantify task-specific trade-offs between model complexity and task performance, as well as the progressive improvement provided by Multi-head Attention (MHA) and Feedforward Networks (FFN). Our results show that applying pruning at the level of mappings instead of individual weights allows for better model compression, when combined with the appropriate regularization. This suggests that developing more global pruning strategies may be a fruitful avenue for future research.

In the future, we plan to apply a similar approach to further reduce the width of Transformer layers, i.e., the hidden dimension, to achieve an even higher compression ratio. We are also interested in jointly using the proposed approach with other compression methods.

Acknowledgement

References

Jimmy Ba and Rich Caruana. 2014. Do deep nets really need to be deep? In *Advances in neural information processing systems*, pages 2654–2662.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Jens Behrmann, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Joern-Henrik Jacobsen. 2019. Invertible Residual Networks. In *International Conference on Machine Learning*, pages 573–582. ISSN: 1938-7228 Section: Machine Learning.

Daoyuan Chen, Yaliang Li, Minghui Qiu, Zhen Wang, Bofang Li, Bolin Ding, Hongbo Deng, Jun Huang, Wei Lin, and Jingren Zhou. 2020. Adabert: Task-adaptive bert compression with differentiable neural architecture search. *arXiv preprint arXiv:2001.04246*.

Zihan Chen, Hongbo Zhang, Xiaoji Zhang, and Leqi Zhao. 2018. Quora question pairs.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. 2017. Parseval networks: Improving robustness to adversarial examples. *arXiv preprint arXiv:1704.08847*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

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*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Angela Fan, Edouard Grave, and Armand Joulin. 2020. Reducing Transformer Depth on Demand with Structured Dropout. In *International Conference on Learning Representations*.

Farzan Farnia, Jesse Zhang, and David Tse. 2018. Generalizable adversarial training via spectral normalization. In *International Conference on Learning Representations*.

Prakhar Ganesh, Yao Chen, Xin Lou, Mohammad Ali Khan, Yin Yang, Deming Chen, Marianne Winslett, Hassan Sajjad, and Preslav Nakov. 2020. Compressing large-scale transformer-based models: A case study on bert. *arXiv preprint arXiv:2002.11985*.

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*.

Fu-Ming Guo, Sijia Liu, Finlay S Mungall, Xue Lin, and Yanzhi Wang. 2019. Reweighted proximal pruning for large-scale language representation. *arXiv preprint arXiv:1909.12486*.

Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016a. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016b. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer.

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

Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. 2016. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657.

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*.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of bert. *arXiv preprint arXiv:1908.08593*.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew Peters, and Noah A Smith. 2019a. Linguistic knowledge and transferability of contextual representations. *arXiv preprint arXiv:1903.08855*.

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

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, pages 14014–14024.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. 2018. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. 2017. Exploring generalization in deep learning. In *Advances in neural information processing systems*, pages 5947–5956.

Adam M Oberman and Jeff Calder. 2018. Lipschitz regularized deep neural networks converge and generalize. *arXiv preprint arXiv:1808.09540*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

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*.

Victor Sanh, Thomas Wolf, and Alexander M Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. *arXiv preprint arXiv:2005.07683*.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2019. Q-bert: Hessian based ultra low precision quantization of bert. *arXiv preprint arXiv:1909.05840*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Jure Sokolić, Raja Giryes, Guillermo Sapiro, and Miguel RD Rodrigues. 2017. Robust large margin deep neural networks. *IEEE Transactions on Signal Processing*, 65(16):4265–4280.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv:1505.00387*.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*.

Gongbo Tang, Rico Sennrich, and Joakim Nivre. 2018. An analysis of attention mechanisms: The case of word sense disambiguation in neural machine translation. *arXiv preprint arXiv:1810.07595*.

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

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*.

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 *Advances in neural information processing systems*, pages 5998–6008.

Andreas Veit and Serge Belongie. 2017. Convolutional networks with adaptive computation graphs. *arXiv preprint arXiv:1711.11503*, 2.

Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-aware neural machine translation learns anaphora resolution. *arXiv preprint arXiv:1805.10163*.

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 *Advances in neural information processing systems*, pages 2074–2082.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Xin Yu, Zhiding Yu, and Srikumar Ramalingam. 2018. Learning strict identity mappings in deep residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4432–4440.

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

Junbo Zhao, Yoon Kim, Kelly Zhang, Alexander Rush, and Yann LeCun. 2018. Adversarially Regularized Autoencoders. In *International Conference on Machine Learning*, pages 5902–5911. ISSN: 1938-7228 Section: Machine Learning.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.
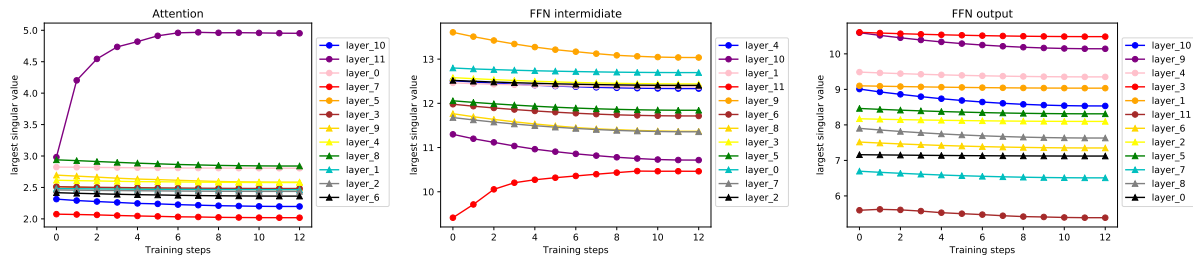
Figure 6: The largest singular value of attention output weights, and FFN intermediate and output weights for the original BERT$_{\text{BASE}}$ during the fine-tuning on the SST-2 task.

Hyperparameter	Pre-training	Fine-tuning
Number of layers	12	12
Hidden size	768	768
FFN inner hidden size	3072	3072
Attention heads	12	12
Attention head size	64	64
Embedding size	768	768
Mask percent	15	-
Learning rate decay	linear	linear
Layerwise LR decay	-	0.8
Warmup steps	10000	-
Warmup fraction	-	0.1
Learning Rate	2e-4	1e-4
Adam ϵ	1e-6	1e-6
Adam β_1	0.9	0.9
Adam	0.999	0.999
Attention dropout	0.1	0.1
Dropout	0.1	0.1
Weight decay	0.01	0.01
l_1 regularization factor	-	0.01
Batch size	2048	32
Train steps	1M	-
Train epochs	-	3.0/iter for MRPC, 1.0 for others

Table 3: Pre-training and Fine-tuning hyperparameters

A   Appendices

A.1   GLUE Dataset

We provide a brief description of the 5 tasks in our experiments from the GLUE benchmarks (Wang et al., 2018).

SST-2   The Stanford Sentiment Treebank (Socher et al., 2013) consists of sentences from movie reviews and human annotations of their sentiment. The task is to predict the sentiment of a given sentence (positive/negative). The performance is evaluated by the accuracy.

QQP   The Quora Question Pairs dataset (Chen et al., 2018) is a collection of question pairs from the community question-answering website Quora. The task is to determine whether a pair of questions are semantically equivalent. The performance is evaluated by the accuracy.

MRPC   The Microsoft Research Paraphrase Corpus (Dolan and Brockett, 2005) is a corpus of sentence pairs automatically extracted from online news sources, with human annotations for whether the sentences in the pair are semantically equivalent, and the task is to predict the equivalence. The performance is evaluated by both the F1 score.

QNLI   The Question-answering NLI dataset (Chen et al., 2018) is converted from the Stanford Question Answering Dataset (SQuAD) to a classification task. The performance is evaluated by the accuracy.

MNLI   The Multi-Genre Natural Language Inference Corpus (Williams et al., 2017) is a crowd-sourced collection of sentence pairs with textual entailment annotations. Given a premise sentence and a hypothesis sentence, the task is to predict whether the premise entails the hypothesis (*entailment*), contradicts the hypothesis (*contradiction*), or neither (*neutral*). The performance is evaluated by the test accuracy on both matched (in-domain) and mismatched (cross-domain) sections of the test data.

A.2   Experiment Settings

The full set of hyperparameters for pre-training and fine-tuning are listed in Table 3.

A.3   Spectral Norm of Weights during Training

We show the largest singular values of the weight metrics in the original BERT model during fine-tuning on the task SST-2 in Figure 6. As can be seen from the figure, the norm of the weights without spectral normalization is obviously out of control.