PAT: Pruning-Aware Tuning for Large Language Models

Yijiang Liu[1], Huanrui Yang[2*], Youxin Chen[3], Rongyu Zhang[1], Miao Wang[1], Yuan Du[1,4], Li Du[1,4*]

[1]School of Electronic Science and Engineering, Nanjing University
[2]University of Arizona
[3]Samsung Electronic Research Centre of China
[4]Interdisciplinary Research Center for Future Intelligent Chips, Nanjing University, Suzhou
{liuyijiang, rongyuzhang, wangmiao}@smail.nju.edu.cn
huanruiyang@arizona.edu, yx113.chen@samsung.com, {yuandu, ldu}@nju.edu.cn

Abstract

Large language models (LLMs) excel in language tasks, especially with supervised fine-tuning after pre-training. However, their substantial memory and computational requirements hinder practical applications. Structural pruning, which reduces less significant weight dimensions, is one solution. Yet, traditional post-hoc pruning often leads to significant performance loss, with limited recovery from further fine-tuning due to reduced capacity. Since the model fine-tuning refines the general and chaotic knowledge in pre-trained models, we aim to incorporate structural pruning with the fine-tuning, and propose the `Pruning-Aware Tuning` (PAT) paradigm to eliminate model redundancy while preserving the model performance to the maximum extend. Specifically, we insert the innovative Hybrid Sparsification Modules (HSMs) between the Attention and FFN components to accordingly sparsify the upstream and downstream linear modules. The HSM comprises a lightweight operator and a globally shared trainable mask. The lightweight operator maintains a training overhead comparable to that of LoRA, while the trainable mask unifies the channels to be sparsified, ensuring structural pruning. Additionally, we propose the Identity Loss which decouples the transformation and scaling properties of the HSMs to enhance training robustness. Extensive experiments demonstrate that PAT excels in both performance and efficiency. For example, our Llama2-7b model with a 25% pruning ratio achieves 1.33× speedup while outperforming the LoRA-finetuned model by up to 1.26% in accuracy with a similar training cost.

Code — https://github.com/kriskrisliu/PAT

Introduction

Large language models (LLMs) (Touvron et al. 2023a; Brown et al. 2020; Chowdhery et al. 2022) have transformed the field of NLP (Vaswani et al. 2017; Bahdanau, Cho, and Bengio 2014; Zhang, Zhao, and LeCun 2015; Yang et al. 2016) with their exceptional performance on various complex language benchmarks. Despite their success, these models often necessitate substantial computational resources and present challenges for practical deployment due to their billions of parameters. Their extensive scales result in high latency and complications in deployments (Pan et al. 2023; Zhang et al. 2024). To mitigate these issues, various techniques have been proposed, including model pruning (Ma, Fang, and Wang 2023; Ashkboos et al. 2024; Sun et al. 2023; Santacroce et al. 2023; Fang, Ma, and Wang 2023), knowledge distillation (Agarwal et al. 2024; Tunstall et al. 2023; Sun et al. 2019, 2020; Ma et al. 2020), and quantization (Liu et al. 2022; Yao et al. 2022; Bai et al. 2020; Zafrir et al. 2019) within the context of pre-trained language models (PLMs).

Network pruning (Syed, Guo, and Sundarapandiyan 2023; Xu et al. 2021a; Liu et al. 2021; Guo et al. 2019), which reduces model size by eliminating specific weights, has gained significant attention. Especially for structural pruning (Ashkboos et al. 2024; Li et al. 2016; Wang et al. 2019b) which promises practical acceleration on current hardware architectures. However, as shown in Fig. 1, traditional pruning methods (Ma, Fang, and Wang 2023; Ashk-

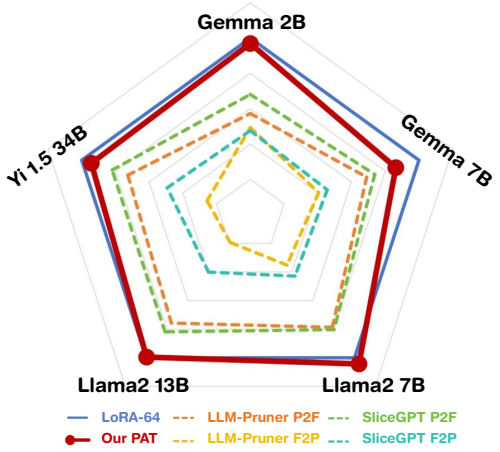


Figure 1: Comparison of zero-shot accuracy averaged on downstream tasks. Various pruning methods at a 25% pruning ratio, as well as the unpruned LoRA, are employed. Our PAT (red) notably outperforms LLM-Pruner and SliceGPT, and is comparable to LoRA (blue), surpassing LoRA by 1.26% on the Llama2-7B model.

*Corresponding author.

boos et al. 2024) usually results in significant performance loss, whether applied before or after recovery model finetuning with Pre/Post-Trainig Pruning (P2F/F2P).

On the other hand, since the pretraining-fine-tuning pipeline has become standard practice in both academic and industrial scenarios, Parameter-Efficient Fine-Tuning (PEFT) methods (Xu et al. 2023a; Lin, Madotto, and Fung 2020; Mahabadi et al. 2021; Liu et al. 2024b), e.g., Low-Rank Adapter (LoRA) (Hu et al. 2021), have emerged as prevailing solutions for streamlined training. Meanwhile, since model fine-tuning can be seen as refining the universal and chaotic knowledge in the pre-trained model, thereby transforming the general LLM into a task-specific expert, combining structural pruning and PEFT for model efficiency and quick adaptation becomes a natural thought.

Drawing inspiration from quantization methods that often work synergistically, including the training-free Post-Training Quantization (PTQ) (Dettmers et al. 2022; Frantar et al. 2022; Lin et al. 2023; Lee et al. 2023) and the performance-enhancing Quantization-Aware Training (QAT) (Liu et al. 2023; Kim et al. 2023; Dettmers et al. 2023), we aim to incorporate structure pruning into the fine-tuning process while further boosting the model performance. This prompts us to introduce a new `Pruning-Aware Tuning` (PAT) paradigm to facilitate efficient inference and practical deployment in real-world applications, such as autonomous vehicles which require fast and accurate model inference to make real-time decisions and avoid obstacles while a fine-tuned RAG model must quickly and precisely retrieve and generate relevant responses from a compact knowledge base for different customer support. Unlike traditional P2F/F2P methods that remove model weights based on fixed prior knowledge, our proposed PAT method enables simultaneous pruning and fine-tuning. This allows the model to adaptively learn which parameters are most redundant and should be pruned during the PAT process. As a result, we achieve an automatic, end-to-end structured pruning process that not only maximizes but can also enhance the capabilities of the fine-tuned model.

Specifically, we propose the integration of plug-in Hybrid Sparsification Modules (HSMs). These HSMs are strategically positioned between the Attention and FFN components. Initially, they are set as identity matrices to maintain stable gradients at the onset of the fine-tuning process. As fine-tuning progresses, the HSMs selectively attenuate the channel values of the hidden dimensions, resulting in the exclusion of the corresponding linear projection weights. However, directly integrating dense-structured HSMs introduces an excess of trainable parameters. To mitigate this issue, we leverage the Hybrid-Identity-Operator (HIO), which reduces the number of trainable parameters. Compared with other PEFT methods, our approach not only achieves parameter efficiency but also decreases the overall model complexity. Furthermore, we introduce the Identity Loss (IL) applied to the HSMs to enhance training robustness and efficacy. This technique regularizes the HSMs while delegating the scaling functionality to independent trainable parameters.

In addition, the pruning operation across all HSMs is governed by a single trainable Unified Sparsification Mask (USM), ensuring consistent retention of channel indices across modules. This approach standardizes and streamlines the transformer decoder structure. As the trainable mask gradually converges to the target sparsity, the knowledge encoded in weights from pruned channels are seamlessly updated and redistributed to the remaining active channels.

Extensive experiments on widely recognized Large Language Models (LLMs) demonstrate the effectiveness of our proposed `Pruning-Aware Tuning` (PAT) compared to state-of-the-art baselines, including Parameter-Efficient Fine-Tuning (PEFT) and Pre/Post-Training Pruning (PTP) methods. Notably, on the Llama2-7B model, PAT surpasses the performance of LoRA-64 by 1.26% while achieving 25% weight pruning. The contribution of this paper can be summarized as follows:

- We propose an innovative paradigm called `Pruning-Aware Tuning` (PAT). Unlike traditional pre- or post-training pruning methods, PAT achieves simultaneous structural pruning and fine-tuning, leading to improved model performance.
- To decrease overall model complexity, we integrate plug-in Hybrid Sparsification Modules (HSMs) with the Hybrid-Identity-Operator. Additionally, we design an Identity Loss (IL) applied to the HSMs to further enhance fine-tuning efficiency and robustness.
- We utilize a single Unified Sparsification Mask (USM) that governs all HSMs, ensuring consistent retention of channel indices across modules.

Related Work

Pruning

Network pruning (LeCun, Denker, and Solla 1989) has long been recognized as an effective method for model compression and acceleration. Earlier research primarily focused on small-scale networks (Fang et al. 2023; Yang et al. 2023; Chen et al. 2021; Wu et al. 2024). However, with the advent of large-scale models, pruning techniques have increasingly been applied to large language models (LLMs). According to the pruning granularity, pruning methods can be broadly categorized into unstructured and structured pruning. In the realm of unstructured pruning (Frantar and Alistarh 2023; Sun et al. 2023), techniques such as SparseGPT (Frantar and Alistarh 2023) and Wanda (Sun et al. 2023) have been proposed. SparseGPT addresses the layer-wise reconstruction problem by utilizing Hessian inverses, while Wanda employs the product of weight magnitudes and input feature norms as its pruning criterion. Despite their effectiveness, these unstructured sparsification methods do not guarantee on-device speedup without hardware-specific support. In contrast, the structured pruning (Zafrir et al. 2021; Kurtic et al. 2022; Xia, Zhong, and Chen 2022; Yang, Wen, and Li 2019; Yang et al. 2023) removes organized patterns within the network, enabling significant acceleration in a hardware-agnostic manner. For instance, Shortened-LLaMA (Kim et al. 2024) removes Transformer blocks, resulting in depth pruning. Sheared-LLaMA (Xia et al. 2023)

incorporates the learnable mask to prune both the network's width and depth. LLM-Pruner (Ma, Fang, and Wang 2023) and SliceGPT (Ashkboos et al. 2024) prune the network width while retaining the number of layers: LLM-Pruner sparsifies the intermediate dimension while SliceGPT focuses on the hidden dimension. However, existing structured pruning models still suffer from accuracy loss, necessitating further exploration and improvement.

Parameter-Efficient Fine-Tuning

Compared to full fine-tuning of LLMs, Parameter-Efficient Fine-Tuning (PEFT) can achieve comparable performance while significantly reducing the computation and memory cost. PEFT methods can be broadly classified into five categories: additive fine-tuning, partial fine-tuning, reparameterized fine-tuning, hybrid fine-tuning, and unified fine-tuning. Additive fine-tuning methods introduce new additional parameters into the model, including adapter-based (Hu et al. 2021; Zhang et al. 2023b; He et al. 2021; Rücklé et al. 2020) and soft prompt-based (Li and Liang 2021; Wang et al. 2023; Vu et al. 2021) approaches. For example, LoRA (Hu et al. 2021), one of the most popular used PEFT method, freezes the pre-trained model weights and injects trainable rank decomposition matrices into each layer of the Transformer architecture, greatly reducing the number of trainable parameters for downstream tasks. DoRA (Liu et al. 2024a), a successful variant of LoRA, achieves enhanced performance by decomposing the pre-trained weights into magnitude and direction for subsequent fine-tuning. Partial fine-tuning selects only the parameters that are important for the downstream task to be trained (Ben-Zaken, Ravfogel, and Goldberg 2021; Lawton et al. 2023; Xu et al. 2021b). Reparameterized fine-tuning methods (Edalati et al. 2022; Zhang et al. 2023a; Xu et al. 2023b) often use low-rank transformations to reduce the number of trainable parameters. Hybrid fine-tuning (Zhou et al. 2023; Hu et al. 2022) combines multiple PEFT methods together. Unified fine-tuning (He et al. 2022; Wang et al. 2022) integrates various fine-tuning methods into a unified structure, but only utilizes one of them during fine-tuning. In this study, we mainly employ LoRA and DoRA as the fine-tuning techniques to explore our proposed PAT paradigm.

Methodology

In this section, we detail the components of our proposed `Pruning-Aware Tuning` (PAT). Firstly, we introduce the foundational concept of the zero-preservation property inherent in the RMSNorm operation. Subsequently, we elaborate on the Hybrid Sparsification Module (HSM) and the Unified Sparsification Mask (USM). Furthermore, we outline the comprehensive process of PAT and introduce the innovative Identity Loss (IL). Finally, we expound on the overall optimization objective.

Preliminary: Zero-Preservation of RMSNorm

RMSNorm (Zhang and Sennrich 2019), an abbreviation for root mean square layer normalization, is widely used in LLMs, such as Llama (Touvron et al. 2023b), Gemma (Team et al. 2024), and Yi (Young et al. 2024). The general form of the RMSNorm is defined as the following:

$$\bar{x}_i = \mathrm{RMSNorm}(x_i) = \frac{x_i}{\mathrm{RMS}(\mathbf{x})} g_i, \tag{1}$$

where $\bar{x}_i$ is the i-th value of vector $\bar{\mathbf{x}} \in \mathbb{R}^d$, and $\mathbf{g} \in \mathbb{R}^d$ is the gain parameter. $\mathrm{RMS}(\cdot)$ is the Root Mean Square operation, defined as:

$$\mathrm{RMS}(\mathbf{x}) = \sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2} \tag{2}$$

Given the layer input $\mathbf{X} \in \mathbb{R}^{d\times n}$ with specific (e.g., 1st and 2nd) channels all equal to $\mathbf{0}$:

$$\mathbf{X} = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ x_3^{(1)} & x_3^{(2)} & \cdots & x_3^{(n)} \\ \vdots & \vdots & \ddots & \vdots \\ x_d^{(1)} & x_d^{(2)} & \cdots & x_d^{(n)} \end{pmatrix} \tag{3}$$

where $x_j^{(i)}$ is the j-th value of the i-th vector in $\mathbf{X}$. Referring to Eq. (1), the RMSNorm operation will preserve these zero values, thereby making it feasible to prune the corresponding channels.

Hybrid Sparsification Module (HSM)

Our objective is to prune the hidden dimensions of LLMs during fine-tuning, which would involve selecting the channels to be pruned in a linear layer, and convert the knowledge of pruned weights into those remained. To achieve this, we design a specific module to be applied after a linear layer, namely Hybrid Sparsification Module (HSM). HSM consists of a trainable channel selection mask $\mathbf{M}$ and a knowledge transformation weight $\mathbf{D}$. Specifically, the computation involving the HSM and the upstream linear layer with weight $\mathbf{W} \in \mathbb{R}^{d_o\times d_i}$ is formulated as follows:

$$\begin{aligned} \mathbf{Z} &= (\mathbf{M} \odot \mathbf{D}) \cdot \mathbf{W}\mathbf{X} \\ &= (\mathbf{M} \odot \mathbf{D}\mathbf{W}) \cdot \mathbf{X} \\ &= \mathbf{W}_D \cdot \mathbf{X}, \end{aligned} \tag{4}$$

where d_i and d_o are the input and output dimension, respectively, $\mathbf{X} \in \mathbb{R}^{d_i\times n}$ is the input value, $\mathbf{Z} \in \mathbb{R}^{d_o\times n}$ is the output value, $\mathbf{M} \in \mathbb{R}^{d_o}$ denotes the trainable mask whose values converge to either 0 or 1, $\mathbf{D} \in \mathbb{R}^{d_o\times d_o}$ is the HSM weight, $\mathbf{W} \in \mathbb{R}^{d_o\times d_i}$ is the upstream linear weight, and $\mathbf{W}_D \in \mathbb{R}^{d_o\times d_i}$ is the merged weight that replaces $\mathbf{W}$ after training. Notably, the zero values in $\mathbf{M}$ effectively cause the corresponding output channels of $\mathbf{W}_D$ to be pruned.

To prune all linear layers in LLMs such as Llama2, which encompass the Q, K, V, and O projections in Attentions, as well as Up, Gate, and Down projections in FFNs, a straightforward approach is to apply the HSM after all linear layers. However, considering the sheer number of the linear layers in an LLM, this approach would incur significant overhead. We propose a novel and efficient alternative: placing pruning modules only between the Attention and FFN

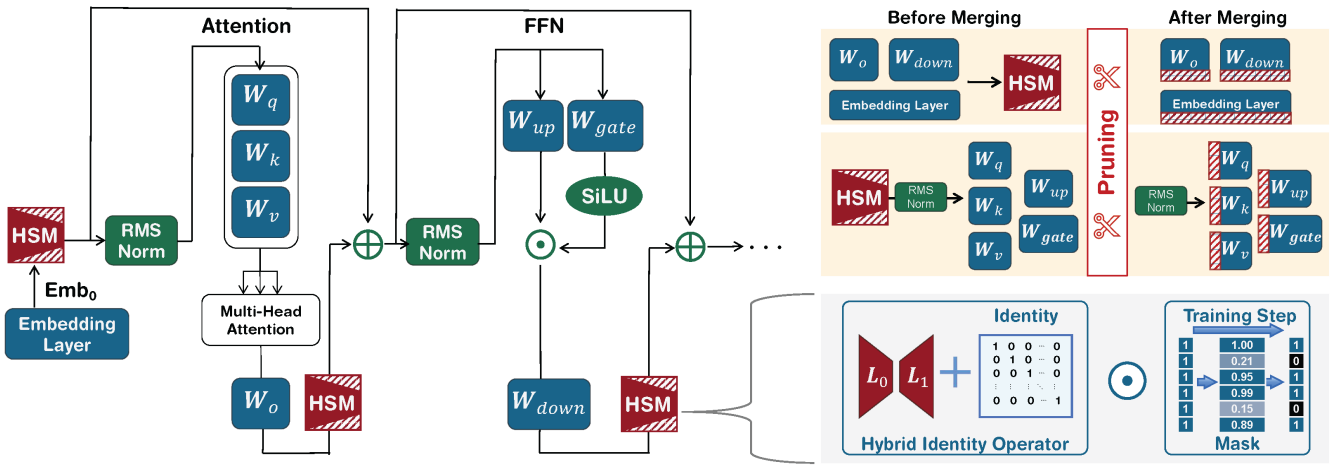


Figure 2: Framework of our Pruning-Aware Tuning (PAT), featuring Hybrid Sparsification Modules (HSMs) positioned between the Attention and Feed-Forward Network (FFN) components. Each HSM includes a Hybrid-Identity-Operator (HIO) and a globally shared trainable mask. At training stage, the mask values will be updated until convergence. At inference stage, the pruned HSMs and the upstream linear layers will be merged, and the downstream layers which receive inputs with zero-valued channels will be pruned accordingly.

components, as illustrated in Fig. 2. The "pruned[1]" HSM's output, $\mathbf{Z}$, will first undergo the addition with the residual connection, which has already been pruned by the previous HSM, and then be fed into the RMSNorm operator before the next Attention/FFN component. As demonstrated previously in the preliminary, the RMSNorm has no impact on zero-valued channels, and since the downstream linear projection receives input with certain channels set to zero, the input dimensions of the following block can be pruned accordingly. In cases where LLMs involve the LayerNorm which projects zero-valued channels to non-zero, we can convert it to the RMSNorm before incorporating HSMs. This transformation is mathematically equivalent, as described by SliceGPT (Ashkboos et al. 2024).

Although inserting HSMs between Attention and FFN components reduces trainable parameters compared to directly applying them to each linear module, the overall training overhead remains significantly larger than that of PEFT methods. To mitigate this issue, we propose the Hybrid-Identity-Operator (HIO) as a replacement for the dense structure of HSMs, which is formulated as:

$$\mathbf{D} = \mathbf{L}_1 \cdot \mathbf{L}_0 + \mathbf{I}, \tag{5}$$

where $\mathbf{L}_0 \in \mathbb{R}^{r \times d_o}$, $\mathbf{L}_1 \in \mathbb{R}^{d_o \times r}$, r is the rank value of $\mathbf{L}_1\mathbf{L}_0$, and $\mathbf{I} \in \mathbb{R}^{d_o \times d_o}$ is the identity matrix with diagonal values set to 1 and other values set to 0. During fine-tuning, $\mathbf{I}$ is frozen, allowing gradients to flow through $\mathbf{L}_0$ and $\mathbf{L}_1$. HIO significantly reduces the number of trainable parameters. For example, a dense HSM consists of $d_o \times d_o$ parameters, while the HIO consists of $2 \times d_o \times r$. By determining $r < d_o/2$, we can decrease the number of trainable parameters. In practice, we set r to approximately 5% of d, which in turn only accounts for 10% parameter of dense HSMs.

[1]At this point, we indicate the zero-valued channels as pruned ones to explain the feasibility of pruning in downstream computations.

Unified Sparsification Mask (USM)

We utilize a single trainable mask M as in Eq. (4) to adaptively set channel values of hidden states to zero. The mask acts uniformly across all HSMs, ensuring consistency in the pruned channel indices throughout the computation flow. This unified pruning mask is particularly necessary at the residual connections between Attention and FFN components, as it guarantees that the pruned channels are correctly aligned throughout the entire data flow.

To insure structural sparsity at the convergence of the model, we employ a continuous sparsification strategy with a tailored regularizer to ensure that the mask converges to discrete values of 0 or 1 and achieves the desired sparsity at the end of the training process. This involves applying a differentiable gating function, $\mathcal{G}(\cdot)$, to the trainable proxy weights $\mathbf{W}_M$ of the mask. The gating function utilizes a modified Sigmoid function with a variable temperature τ, which is defined as:

$$\tau(s) = \begin{cases} \dfrac{1}{1 - \dfrac{\ln(s)}{\ln(s_0)}} & \text{if } s < s_0, \\ \epsilon^{-1} & \text{otherwise.} \end{cases} \tag{6}$$

$$\beta(s) = \begin{cases} \dfrac{-s}{s_0} + 0.5 & \text{if } s < s_0/2, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

$$\mathbf{M} = \mathcal{G}(s, \mathbf{W}_M) = \frac{1}{1 + e^{-\tau(s) \cdot \mathbf{W}_M}} + \beta(s), \tag{8}$$

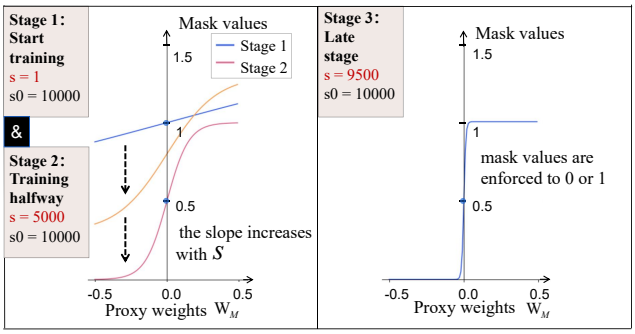


Figure 3: The differentiable gating function $\mathcal{G}(\cdot)$.

where s denotes the current training step which dynamically determines the temperature, s_0 is the milestone step which indicates that the temperature stay unchanged in the remaining training steps. In practice, we set s_0 to $1/3$ of the total training steps. $\beta(\cdot)$ denotes the offset which varies according to the step. Fig. 3 demonstrates some typical training stages. Initially, when $s = 0$, the gating function maps all proxy weights of the mask to 1. This is achieved by initializing $\mathbf{W}_M$ to zero, which keeps the model weights unchanged, ensuring stable gradients at the beginning. As the temperature increases, the slope near 0 rises, and the offset term decreases. By halfway to the milestone step, the offset term reaches 0 and stops updating, while the slope continues to increase. At the milestone step, the slope near 0 becomes very steep, while the slope elsewhere approaches 0. At this point, the mask values will be enforced to either 0 or 1, where 0 refers to the channel being pruned. Moreover, to achieve the target sparsity, specifically the proportion of values equal to 0, we propose regularizing the number of active channels. This is achieved through the following regularization term:

$$\mathcal{L}_{active} = \|N_{target} - \sum_i \mathbb{1}_{(m_i>0)}\|_2, \tag{9}$$

where N_{target} denotes the target channel number of active channels, m_i represents the i-th value of the proxy weight M, and $\mathbb{1}_{(condition)}$ is the indicator function that equals 1 if the condition is true, and 0 otherwise.

Pruning-Aware Tuning

We perform model fine-tuning by updating the proposed HSM modules and applying LoRA on all linear layers (Hu et al. 2021). Besides the standard instruction fine-tuning loss $\mathcal{L}_{Instruct}$, we propose the innovative Identity Loss (IL) to decompose the scaling and rotation in the HSM transformations. Specifically, we alter the formulation of Eq. (5) into:

$$\mathbf{D} = \mathbf{L}_1 \cdot \mathrm{diag}(\mathbf{v}) \cdot \mathbf{L}_0 + \mathbf{I}, \tag{10}$$

where $\mathbf{v} \in \mathbb{R}^r$ is the trainable scaling values, and L_0 and L_1 are constrained to be orthogonal with the identity regularization

$$\mathcal{L}_{Identity} = \|\mathbf{L}_0 \cdot \mathbf{L}_0^T - \mathbf{I}\|_2 + \|\mathbf{L}_1^T \cdot \mathbf{L}_1 - \mathbf{I}\|_2 \tag{11}$$

The overall optimization objective is defined by a composite loss function $\mathcal{L}$, which is expressed as follows:

$$\mathcal{L} = \mathcal{L}_{Instruct} + \mathcal{L}_{active} + \mathcal{L}_{Identity}, \tag{12}$$

where $\mathcal{L}_{Instruct}$ represents the loss associated with instruction fine-tuning.

Experiments

In this section, we present the experimental results and analysis. We begin by describing the experimental setup. Next, we showcase our main results across various Language Models (LLMs). We then delve into the efficiency and accuracy trade-off, examining memory and latency considerations. Finally, we conduct ablation studies on the trainable mask and identity loss.

Experimental Setup

Models. We utilize model frameworks and checkpoints from HuggingFace (Jain 2022; Wolf et al. 2019), which includes Llama-2 7B and 13B (Touvron et al. 2023b), Gemma 2B and 7B (Team et al. 2024), Yi-1.5-34B (Young et al. 2024).

Baselines. The pruning baselines include LLM-Pruner (Ma, Fang, and Wang 2023), and SliceGPT (Ashkboos et al. 2024). We also involve the common LoRA (Hu et al. 2021) approach with the rank set to 64. Unless otherwise stated, we adjust the number of trainable parameters in all fine-tuning approaches to match the number of the LoRA. Additionally, we conduct complementary tests by applying "P→FT" (Pruning before Fine-Tuning) and "FT→P" (Fine-Tuning before Pruning) strategies on LLM-Pruner and SliceGPT. The pruning ratios are set to 20%, 25%, and 30%, respectively.

Datasets. We employ the LaMini-instruction dataset (Wu et al. 2023) for fine-tuning. To reduce training costs, we randomly drop 50% of the samples, resulting in a final dataset of 1 million samples. Unless otherwise stated, all experimental results are based on this setting. We conduct zero-shot evaluation on 14 datasets, including ARC-Challenge (Clark et al. 2018), ARC-Easy (Clark et al. 2018), BOOLQ (Wang et al. 2019a), COPA (Wang et al. 2019a), HellaSwag (Zellers et al. 2019), MMLU (Hendrycks et al. 2021), MultiRC (Wang et al. 2019a), OpenBookQA (Mihaylov et al. 2018), PIQA (Bisk et al. 2020), RTE (Wang et al. 2019a), SIQA (Sap et al. 2019), WIC (Wang et al. 2019a), WinoGrande (Sakaguchi et al. 2021), WSC (Wang et al. 2019a). The accuracy is calulated by First-Capital-Word[2] (Contributors 2023) method.

Implementation Details. Experiments are conducted using A100 GPUs. The models are fine-tuned over 3 epochs using the Alpaca instruction template. The learning rate is set to 5×10^{-5} with a cosine schedule. The batch size is set to 128, and the sequence length is 256 tokens. The milestone step of our PAT, s_0, is set to $1/3$ of the total training steps. The settings of our HIOs are derived to match the number of trainable parameters with LoRA-64. For example, we set the rank values of HIO and LoRA modules to 200 and 20 in the Llama2-7B experiments, respectively.

[2]https://github.com/open-compass/opencompass

Experimental Results and Analysis

Performance Comparison. Tab. 1 shows the zero-shot evaluations of different pruning methods across 14 well-known tasks, where various types and sizes of LLMs are tested. We obtain that: (1) Our method, employing the Pruning-Aware Tuning (PAT) strategy, achieves the highest accuracy across pruned models. In contrast, LLM-Pruner and SliceGPT, which use either the Pruning before Fine-Tuning (P→FT) or Fine-Tuning before Pruning (FT→P), suffer from non-negligible accuracy degradation. However, the"P→FT" significantly outperforms the "FT→P". (2) The feasibility of pruning varies across different models. We observe that Llama2 with PAT maintains comparable performance to the un-pruned LoRA approach even at a 30% pruning rate, whereas Gemma 7b shows the trending of accuracy degradation at a 20% pruning rate. (3) Surprisingly, Llama2 7B and 13B with PAT under less than 30% and 20% pruning ratio, respectively, exhibit accuracy better than the un-pruned LoRA.

Efficiency and Accuracy Trade-off. The implementation of HIO significantly reduces the number of trainable parameters, but this reduction may directly impact the model accuracy. We conducted experiments using various scales of training parameters on the Llama 2 7B model, and illustrate the results in Fig. 4. The total number of trainable parameters is adjusted by the rank values of HIO and LoRA modules. For example, the "LoRA-64" in dark represents the traditional LoRA fine-tuning with a rank value set to 64, and the "HIO-200, LoRA-20" in purple represents our PAT with a rank of 200 in HIO and a rank of 20 in LoRA modules. We find that our PAT demonstrates a performance trend correlated to the number of trainable parameters. "Dense[3], LoRA-8" with 14.15% trainable parameters achieves 64.19% accuracy, outperforming "LoRA-64" by 5.43%. Conversely, "HIO-8, LoRA-8" with merely 0.36% trainable parameters results in a 6% accuracy reduction. In practice, we opt for "HIO-200, LoRA-20" in Llama 2 7B experiments, aligning the parameter count with that of "LoRA-64". For others, Gemma 2B with "HIO-300, LoRA20", Gemma 7B with "HIO-300, LoRA20", Llama2 13B with "HIO-200, LoRA20" , and Yi-1.5 34B with "HIO-200, LoRA20".

Memory and Latency. We conducted an evaluation of the VRAM usage and the inference latency comparing the base Llama2 7B and 13B models with pruned versions, as illustrated in Fig. 5 and Fig. 6. The GPU memory is tested by loading the model without any proceeding tokens. The latency is tested by the time of the first token prediction in a batch with an initial context length of 128. Specifically, we assessed the models pruned at 20%, 25%, and 30% ratios across various batch sizes. Our 30% pruned models achieve 1.33× speedup on average. Moreover, the base Llama2 13B model encounters Out-Of-Memory (OOM) errors at a batch size of larger than 288 when executed on a single A100-80GB GPU. In contrast, our pruned models work reliably under these conditions.

[3]Indicating that we use the dense matrix instead of the HIO.

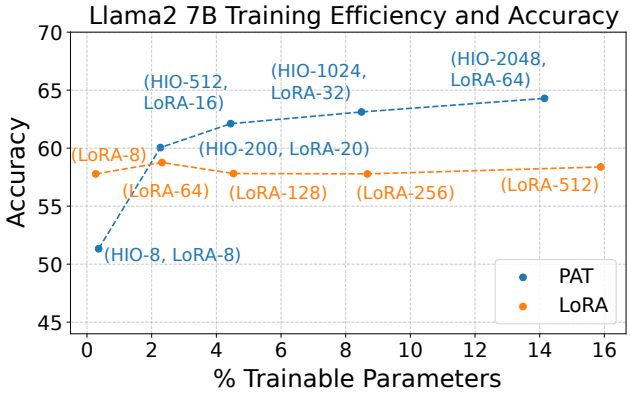
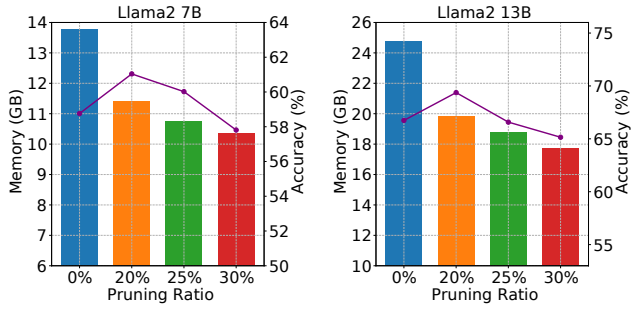
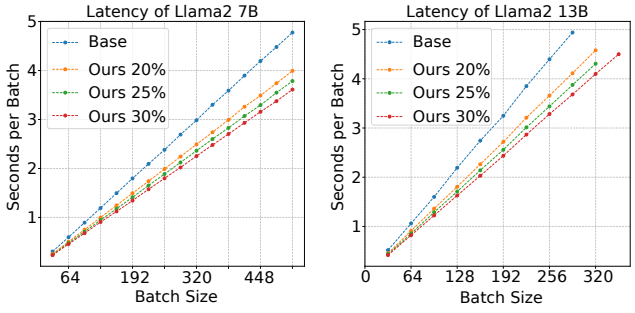


Figure 4: The training efficiency and the accuracy comparison for Llama2 7B. Our PAT results are represented as "HIO-M, LoRA-N", where M and N denote the rank value in the HIO and the LoRA, respectively. The LoRA results are "LoRA-N".



(a) Llama 2 7B (b) Llama 2 13B

Figure 5: The VRAM usage and the evaluation accuracy of Llama2 models under various pruning ratios.



(a) Llama 2 7B (b) Llama 2 13B

Figure 6: The speedup of Llama2 models according to different pruning ratios and batch sizes.

Ratio	Method	Mode	Gemma-2B	Gemma-7B	Llama2-7B	Llama2-13B	Yi-1.5-34B
0%	**LoRA-64**	FT	53.82	71.59	58.76	66.74	81.21
20%	LLM-Pruner	P→FT	48.87	65.45	58.53	65.28	73.86
	LLM-Pruner	FT→P	40.64	54.87	40.68	41.43	53.88
	SliceGPT	P→FT	48.21	66.60	57.81	65.86	76.81
	SliceGPT	FT→P	41.67	56.17	47.77	50.67	67.60
	Ours	**PAT**	**53.95**	**68.68**	**61.04**	**69.37**	**81.02**
25%	LLM-Pruner	P→FT	42.32	60.50	52.50	58.64	70.10
	LLM-Pruner	FT→P	40.20	50.29	39.72	39.82	51.03
	SliceGPT	P→FT	45.23	62.22	52.98	60.69	73.88
	SliceGPT	FT→P	39.72	52.13	41.97	46.75	60.63
	Ours	**PAT**	**52.98**	**66.68**	**60.02**	**66.58**	**78.90**
30%	LLM-Pruner	P→FT	39.71	50.28	50.60	51.28	66.85
	LLM-Pruner	FT→P	40.05	41.35	39.73	39.70	45.36
	SliceGPT	P→FT	40.07	53.14	50.91	56.12	71.81
	SliceGPT	FT→P	39.89	44.30	40.14	46.19	56.10
	Ours	**PAT**	**45.33**	**64.58**	**57.81**	**65.15**	**77.89**

Table 1: Zero-shot evaluations of different pruning methods with 20%, 25%, and 30% pruning ratios across various LLMs. "**FT**" represents **F**ine-**T**uning. "**P**→**FT**" denotes **P**runing the base model and then **F**ine-**T**uning the pruned model via LoRA. "**FT**→**P**" denotes **F**ine-**T**uning the base model via LoRA and then **P**runing the fine-tuned model. "**PAT**" denotes our proposed **P**runing-**A**ware **T**uning strategy. The accuracy is averaged across 14 datasets. More details are available in the Appendix.

Trainable and Frozen Mask. The frozen mask is implemented by linearly attenuating a fixed portion of the mask values during training. In our experiment, this attenuation is applied to the first N values of the hidden dimension in LLMs, where N is determined by the pruning ratio. The results presented in Tab. 2 demonstrate the significant advantage of the trainable mask over the frozen counterpart. For instance, in the case of the Llama2 13B model with 30% pruning, the trainable mask yields an accuracy improvement of 4.06% over the frozen mask.

Model	Ratio	Method	Trainable Mask	Identity Loss	Accuracy
Llama2 7B	0%	LoRA	N/A	N/A	58.76
	25%	PAT	✗	✓	54.97
			✓	✗	58.62
			✓	✓	**60.02**
	30%	PAT	✗	✓	52.72
			✓	✗	56.59
			✓	✓	**57.81**
Llama2 13B	0%	LoRA	N/A	N/A	66.74
	25%	PAT	✗	✓	62.35
			✓	✗	65.81
			✓	✓	**66.58**
	30%	PAT	✗	✓	61.09
			✓	✗	64.85
			✓	✓	**65.15**

Table 2: Ablation study on trainable mask and identity loss.

Ablation on Identity Loss. The incorporation of Identity Loss contributes to an enhanced accuracy improvement. As depicted in Tab. 2, Llama2 7B achieves 1.4% enhancement with the pruning ratio of 25%.

Downstream Task Capability. Following the downstream task adaptation detailed in DoRA (Liu et al. 2024a), we leverage PAT to fine-tune on specific tasks, including ARC, SuperGlue, OpenBookQA, PIQA, SIQA, MMLU, and WinoGrande. The setting of HSMs is "HIO-200, LoRA/DoRA-20". Our 25% pruned PAT-L and PAT-D achieve performance levels on par with those achieved by traditional DoRA and LoRA, shown in Tab. 3.

Method	Ratio	Llama2 7B	Llama2 13B
LoRA-64	0%	72.85	76.24
PAT-L	25%	72.05	76.08
DoRA-64	0%	73.50	77.36
PAT-D	25%	72.98	77.02

Table 3: Downstream task performance of LoRA, DoRA, and PAT. PAT-L and PAT-D denote our PAT with LoRA and DoRA fine-tuning, respectively.

Conclusion

We propose Pruning-Aware Tuning (PAT), a novel structured pruning approach for Large Language Models (LLMs). PAT prunes the hidden dimensions during the fine-tuning, while preserving the linguistic capabilities. We develop a trainable mask to adaptively set channel values to zero, and efficient Hybrid Sparsification Modules to enable pruning of all linear layers accordingly. The efficiency design reduces the training overhead of PAT to levels comparable to traditional LoRA fine-tuning. Additionally, we propose the Identity Loss to enhance the training robustness by decoupling the rotation and scaling properties of the HSMs. In the zero-shot evaluation, our 30%-PAT Llama2 7B and 13B models maintains 98% performance of those achieved from the LoRA fine-tuning.

Acknowledgments

This work was supported in part by the Strategic Industries and Key Technologies Project of Jiangsu Province under Grant BE2023020-3.

References

Agarwal, R.; Vieillard, N.; Zhou, Y.; Stanczyk, P.; Garea, S. R.; Geist, M.; and Bachem, O. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*.

Ashkboos, S.; Croci, M. L.; Nascimento, M. G. d.; Hoefler, T.; and Hensman, J. 2024. Slicegpt: Compress large language models by deleting rows and columns. *arXiv preprint arXiv:2401.15024*.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*, abs/1409.0473.

Bai, H.; Zhang, W.; Hou, L.; Shang, L.; Jin, J.; Jiang, X.; Liu, Q.; Lyu, M. R.; and King, I. 2020. BinaryBERT: Pushing the Limit of BERT Quantization. In *Annual Meeting of the Association for Computational Linguistics*.

Ben-Zaken, E.; Ravfogel, S.; and Goldberg, Y. 2021. BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models. *ArXiv*, abs/2106.10199.

Bisk, Y.; Zellers, R.; Bras, R. L.; Gao, J.; and Choi, Y. 2020. PIQA: Reasoning about Physical Commonsense in Natural Language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. *ArXiv*, abs/2005.14165.

Chen, T.; Ji, B.; Ding, T.; Fang, B.; Wang, G.; Zhu, Z.; Liang, L.; Shi, Y.; Yi, S.; and Tu, X. 2021. Only train once: A one-shot neural network training and pruning framework. *Advances in Neural Information Processing Systems*, 34: 19637–19651.

Chowdhery, A.; Narang, S.; Devlin, J.; and et al. 2022. PaLM: Scaling Language Modeling with Pathways. *J. Mach. Learn. Res.*, 24: 240:1–240:113.

Clark, P.; Cowhey, I.; Etzioni, O.; Khot, T.; Sabharwal, A.; Schoenick, C.; and Tafjord, O. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *arXiv:1803.05457v1*.

Contributors, O. 2023. OpenCompass: A Universal Evaluation Platform for Foundation Models. https://github.com/open-compass/opencompass.

Dettmers, T.; Lewis, M.; Belkada, Y.; and Zettlemoyer, L. 2022. LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale. *ArXiv*, abs/2208.07339.

Dettmers, T.; Pagnoni, A.; Holtzman, A.; and Zettlemoyer, L. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. *ArXiv*, abs/2305.14314.

Edalati, A.; Tahaei, M. S.; Kobyzev, I.; Nia, V.; Clark, J. J.; and Rezagholizadeh, M. 2022. KronA: Parameter Efficient Tuning with Kronecker Adapter. *ArXiv*, abs/2212.10650.

Fang, G.; Ma, X.; Song, M.; Mi, M. B.; and Wang, X. 2023. DepGraph: Towards Any Structural Pruning. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 16091–16101.

Fang, G.; Ma, X.; and Wang, X. 2023. Structural Pruning for Diffusion Models. *ArXiv*, abs/2305.10924.

Frantar, E.; and Alistarh, D. 2023. SparseGPT: Massive Language Models Can Be Accurately Pruned in One-Shot. *ArXiv*, abs/2301.00774.

Frantar, E.; Ashkboos, S.; Hoefler, T.; and Alistarh, D. 2022. GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers. *ArXiv*, abs/2210.17323.

Guo, F.-M.; Liu, S.; Mungall, F. S.; Lin, X.; and Wang, Y. 2019. Reweighted Proximal Pruning for Large-Scale Language Representation. *ArXiv*, abs/1909.12486.

He, J.; Zhou, C.; Ma, X.; Berg-Kirkpatrick, T.; and Neubig, G. 2021. Towards a Unified View of Parameter-Efficient Transfer Learning. *ArXiv*, abs/2110.04366.

He, S.; Ding, L.; Dong, D.; Zhang, M.; and Tao, D. 2022. SparseAdapter: An Easy Approach for Improving the Parameter-Efficiency of Adapters. *ArXiv*, abs/2210.04284.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Hu, J. E.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; and Chen, W. 2021. LoRA: Low-Rank Adaptation of Large Language Models. *ArXiv*, abs/2106.09685.

Hu, S.; Zhang, Z.; Ding, N.; Wang, Y.; Wang, Y.; Liu, Z.; and Sun, M. 2022. Sparse Structure Search for Delta Tuning. In *Neural Information Processing Systems*.

Jain, S. M. 2022. Hugging face. In *Introduction to transformers for NLP: With the hugging face library and models to solve problems*, 51–67. Springer.

Kim, B.-K.; Kim, G.; Kim, T.-H.; Castells, T.; Choi, S.; Shin, J.; and Song, H.-K. 2024. Shortened LLaMA: A Simple Depth Pruning for Large Language Models. *ArXiv*, abs/2402.02834.

Kim, J.; Lee, J. H.; Kim, S.; Park, J.; Yoo, K. M.; Kwon, S. J.; and Lee, D. 2023. Memory-Efficient Fine-Tuning of Compressed Large Language Models via sub-4-bit Integer Quantization. *ArXiv*, abs/2305.14152.

Kurtic, E.; Campos, D. F.; Nguyen, T.; Frantar, E.; Kurtz, M.; Fineran, B.; Goin, M.; and Alistarh, D. 2022. The Optimal BERT Surgeon: Scalable and Accurate Second-Order Pruning for Large Language Models. *ArXiv*, abs/2203.07259.

Lawton, N.; Kumar, A.; Thattai, G.; Galstyan, A. G.; and Steeg, G. V. 2023. Neural Architecture Search for

Parameter-Efficient Fine-tuning of Large Pre-trained Language Models. In *Annual Meeting of the Association for Computational Linguistics*.

LeCun, Y.; Denker, J. S.; and Solla, S. A. 1989. Optimal Brain Damage. In *Neural Information Processing Systems*.

Lee, C.; Jin, J.; Kim, T.; Kim, H.; and Park, E. 2023. OWQ: Lessons learned from activation outliers for weight quantization in large language models. *ArXiv*, abs/2306.02272.

Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; and Graf, H. P. 2016. Pruning Filters for Efficient ConvNets. *ArXiv*, abs/1608.08710.

Li, X. L.; and Liang, P. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. *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)*, abs/2101.00190.

Lin, J.; Tang, J.; Tang, H.; Yang, S.; Dang, X.; and Han, S. 2023. AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration. *ArXiv*, abs/2306.00978.

Lin, Z.; Madotto, A.; and Fung, P. 2020. Exploring Versatile Generative Language Model Via Parameter-Efficient Transfer Learning. In *Findings*.

Liu, S.-Y.; Wang, C.-Y.; Yin, H.; Molchanov, P.; Wang, Y.-C. F.; Cheng, K.-T.; and Chen, M.-H. 2024a. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*.

Liu, Y.; Yang, H.; Dong, Z.; Keutzer, K.; Du, L.; and Zhang, S. 2022. NoisyQuant: Noisy Bias-Enhanced Post-Training Activation Quantization for Vision Transformers. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 20321–20330.

Liu, Y.; Zhang, R.; Yang, H.; Keutzer, K.; Du, Y.; Du, L.; and Zhang, S. 2024b. Intuition-aware Mixture-of-Rank-1-Experts for Parameter Efficient Finetuning. *arXiv preprint arXiv:2404.08985*.

Liu, Z.; Li, F.; Li, G.; and Cheng, J. 2021. EBERT: Efficient BERT Inference with Dynamic Structured Pruning. In *Findings*.

Liu, Z.; Oğuz, B.; Zhao, C.; Chang, E.; Stock, P.; Mehdad, Y.; Shi, Y.; Krishnamoorthi, R.; and Chandra, V. 2023. LLM-QAT: Data-Free Quantization Aware Training for Large Language Models. *ArXiv*, abs/2305.17888.

Ma, X.; Fang, G.; and Wang, X. 2023. LLM-Pruner: On the Structural Pruning of Large Language Models. *ArXiv*, abs/2305.11627.

Ma, X.; Shen, Y.; Fang, G.; Chen, C.; Jia, C.; and Lu, W. 2020. Adversarial Self-Supervised Data Free Distillation for Text Classification. In *Conference on Empirical Methods in Natural Language Processing*.

Mahabadi, R. K.; Ruder, S.; Dehghani, M.; and Henderson, J. 2021. Parameter-efficient Multi-task Fine-tuning for Transformers via Shared Hypernetworks. In *Annual Meeting of the Association for Computational Linguistics*.

Mihaylov, T.; Clark, P.; Khot, T.; and Sabharwal, A. 2018. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. In *EMNLP*.

Pan, J.; Wang, C.; Zheng, K.; Li, Y.; Wang, Z.; and Feng, B. 2023. SmoothQuant+: Accurate and Efficient 4-bit Post-Training WeightQuantization for LLM. *ArXiv*, abs/2312.03788.

Rücklé, A.; Geigle, G.; Glockner, M.; Beck, T.; Pfeiffer, J.; Reimers, N.; and Gurevych, I. 2020. AdapterDrop: On the Efficiency of Adapters in Transformers. In *Conference on Empirical Methods in Natural Language Processing*.

Sakaguchi, K.; Bras, R. L.; Bhagavatula, C.; and Choi, Y. 2021. Winogrande: An adversarial winograd schema challenge at scale. In *AAAI Conference on Artificial Intelligence*.

Santacroce, M.; Wen, Z.; Shen, Y.; and Li, Y.-F. 2023. What Matters In The Structured Pruning of Generative Language Models? *ArXiv*, abs/2302.03773.

Sap, M.; Rashkin, H.; Chen, D.; LeBras, R.; and Choi, Y. 2019. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*.

Sun, M.; Liu, Z.; Bair, A.; and Kolter, J. Z. 2023. A Simple and Effective Pruning Approach for Large Language Models. *ArXiv*, abs/2306.11695.

Sun, S.; Cheng, Y.; Gan, Z.; and Liu, J. 2019. Patient Knowledge Distillation for BERT Model Compression. In *Conference on Empirical Methods in Natural Language Processing*.

Sun, S.; Gan, Z.; Cheng, Y.; Fang, Y.; Wang, S.; and Liu, J. 2020. Contrastive Distillation on Intermediate Representations for Language Model Compression. In *Conference on Empirical Methods in Natural Language Processing*.

Syed, A.; Guo, P. H.; and Sundarapandiyan, V. 2023. Prune and Tune: Improving Efficient Pruning Techniques for Massive Language Models. In *Tiny Papers @ ICLR*.

Team, G.; Mesnard, T.; Hardin, C.; Dadashi, R.; Bhupatiraju, S.; Pathak, S.; Sifre, L.; Rivière, M.; Kale, M. S.; Love, J.; et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; Rodriguez, A.; Joulin, A.; Grave, E.; and Lample, G. 2023a. LLaMA: Open and Efficient Foundation Language Models. *ArXiv*, abs/2302.13971.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Tunstall, L.; Beeching, E.; Lambert, N.; Rajani, N.; Rasul, K.; Belkada, Y.; Huang, S.; von Werra, L.; Fourrier, C.; Habib, N.; Sarrazin, N.; Sanseviero, O.; Rush, A. M.; and Wolf, T. 2023. Zephyr: Direct Distillation of LM Alignment. *ArXiv*, abs/2310.16944.

Vaswani, A.; Shazeer, N. M.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In *Neural Information Processing Systems*.

Vu, T.; Lester, B.; Constant, N.; Al-Rfou, R.; and Cer, D. M. 2021. SPoT: Better Frozen Model Adaptation through Soft Prompt Transfer. *ArXiv*, abs/2110.07904.

Wang, A.; Pruksachatkun, Y.; Nangia, N.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2019a. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. *arXiv preprint 1905.00537*.

Wang, C.; Grosse, R. B.; Fidler, S.; and Zhang, G. 2019b. EigenDamage: Structured Pruning in the Kronecker-Factored Eigenbasis. In *International Conference on Machine Learning*.

Wang, Y.; Mukherjee, S.; Liu, X.; Gao, J.; and Gao, J. 2022. AdaMix: Mixture-of-Adaptations for Parameter-efficient Model Tuning. In *Conference on Empirical Methods in Natural Language Processing*.

Wang, Z.; Panda, R.; Karlinsky, L.; Feris, R. S.; Sun, H.; and Kim, Y. 2023. Multitask Prompt Tuning Enables Parameter-Efficient Transfer Learning. *ArXiv*, abs/2303.02861.

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Wu, M.; Waheed, A.; Zhang, C.; Abdul-Mageed, M.; and Aji, A. F. 2023. Lamini-lm: A diverse herd of distilled models from large-scale instructions. *arXiv preprint arXiv:2304.14402*.

Wu, X.; Gao, S.; Zhang, Z.; Li, Z.; Bao, R.; Zhang, Y.; Wang, X.; and Huang, H. 2024. Auto-Train-Once: Controller Network Guided Automatic Network Pruning from Scratch. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16163–16173.

Xia, M.; Gao, T.; Zeng, Z.; and Chen, D. 2023. Sheared LLaMA: Accelerating Language Model Pre-training via Structured Pruning. *ArXiv*, abs/2310.06694.

Xia, M.; Zhong, Z.; and Chen, D. 2022. Structured Pruning Learns Compact and Accurate Models. *ArXiv*, abs/2204.00408.

Xu, D.; Yen, I. E.-H.; Zhao, J.; and Xiao, Z. 2021a. Rethinking Network Pruning – under the Pre-train and Fine-tune Paradigm. *ArXiv*, abs/2104.08682.

Xu, L.; Xie, H.; Qin, S.-Z. J.; Tao, X.; and Wang, F. L. 2023a. Parameter-Efficient Fine-Tuning Methods for Pre-trained Language Models: A Critical Review and Assessment. *ArXiv*, abs/2312.12148.

Xu, R.; Luo, F.; Zhang, Z.; Tan, C.; Chang, B.; Huang, S.; and Huang, F. 2021b. Raise a Child in Large Language Model: Towards Effective and Generalizable Fine-tuning. *ArXiv*, abs/2109.05687.

Xu, Y.; Xie, L.; Gu, X.; Chen, X.; Chang, H.; Zhang, H.; Chen, Z.; Zhang, X.; and Tian, Q. 2023b. QA-LoRA: Quantization-Aware Low-Rank Adaptation of Large Language Models. *ArXiv*, abs/2309.14717.

Yang, H.; Wen, W.; and Li, H. 2019. Deephoyer: Learning sparser neural network with differentiable scale-invariant sparsity measures. *arXiv preprint arXiv:1908.09979*.

Yang, H.; Yin, H.; Shen, M.; Molchanov, P.; Li, H.; and Kautz, J. 2023. Global vision transformer pruning with hessian-aware saliency. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 18547–18557.

Yang, Z.; Yang, D.; Dyer, C.; He, X.; Smola, A.; and Hovy, E. H. 2016. Hierarchical Attention Networks for Document Classification. In *North American Chapter of the Association for Computational Linguistics*.

Yao, Z.; Aminabadi, R. Y.; Zhang, M.; Wu, X.; Li, C.; and He, Y. 2022. ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers. *ArXiv*, abs/2206.01861.

Young, A.; Chen, B.; Li, C.; Huang, C.; Zhang, G.; Zhang, G.; Li, H.; Zhu, J.; Chen, J.; Chang, J.; et al. 2024. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*.

Zafrir, O.; Boudoukh, G.; Izsak, P.; and Wasserblat, M. 2019. Q8BERT: Quantized 8Bit BERT. *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS Edition (EMC2-NIPS)*, 36–39.

Zafrir, O.; Larey, A.; Boudoukh, G.; Shen, H.; and Wasserblat, M. 2021. Prune Once for All: Sparse Pre-Trained Language Models. *ArXiv*, abs/2111.05754.

Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Zhang, B.; and Sennrich, R. 2019. Root Mean Square Layer Normalization. *ArXiv*, abs/1910.07467.

Zhang, M.; Chen, H.; Shen, C.; Yang, Z.; Ou, L.; Yu, X.; and Zhuang, B. 2023a. Pruning Meets Low-Rank Parameter-Efficient Fine-Tuning. *ArXiv*, abs/2305.18403.

Zhang, R.; Han, J.; Zhou, A.; Hu, X.; Yan, S.; Lu, P.; Li, H.; Gao, P.; and Qiao, Y. J. 2023b. LLaMA-Adapter: Efficient Fine-tuning of Language Models with Zero-init Attention. *ArXiv*, abs/2303.16199.

Zhang, R.; Luo, Y.; Liu, J.; Yang, H.; Dong, Z.; Gudovskiy, D. A.; Okuno, T.; Nakata, Y.; Keutzer, K.; Du, Y.; and Zhang, S. 2024. Efficient Deweahter Mixture-of-Experts with Uncertainty-Aware Feature-Wise Linear Modulation. In *AAAI Conference on Artificial Intelligence*.

Zhang, X.; Zhao, J. J.; and LeCun, Y. 2015. Character-level Convolutional Networks for Text Classification. In *Neural Information Processing Systems*.

Zhou, H.; Wan, X.; Vulic, I.; and Korhonen, A. 2023. AutoPEFT: Automatic Configuration Search for Parameter-Efficient Fine-Tuning. *Transactions of the Association for Computational Linguistics*, 12: 525–542.