EcoSpa: Efficient Transformer Training with Coupled Sparsity

Jinqi Xiao[1], **Cheng Luo**[2], **Lingyi Huang**[1], **Cheng Yang**[1], **Yang Sui**[3], **Huy Phan**[1],

Xiao Zang[1], **Yibiao Ying**[1], **Anima Anandkumar**[2], **Bo Yuan**[1]

[1]Rutgers University [2]California Institute of Technology [3]Rice University

Abstract

Transformers have emerged as the backbone neural network architecture in today's AI applications. Due to their high complexity, sparsifying transformers, at both pre-training and fine-tuning stages, is very attractive for lowering the training and inference costs. In this paper, we propose EcoSpa, an efficient structured sparse training approach for language and vision transformers. Unlike prior works focusing on individual building blocks, EcoSpa fully considers the correlation between the weight matrices and their component rows/columns, and performs the coupled estimation and coupled sparsification. To achieve that, EcoSpa introduces the use of new granularity when calibrating the importance of structural components in the transformer and removing the insignificant parts. Evaluations across different models, in both pre-training and fine-tuning scenarios, demonstrate the effectiveness of the proposed approach. EcoSpa leads to $2.2\times$ size reduction with 2.4 lower perplexity when training GPT-2 model from scratch. It also enables $1.6\times$ training speedup over the pre-training method. For training sparse LLaMA-1B from scratch, our approach reduces GPU memory usage by 50%, decreases training time by 21%, and achieves a $1.6\times$ speedup in inference throughput while maintaining model performance. Experiments of applying EcoSpa for fine-tuning tasks also show significant performance improvement with respect to model accuracy and pruning cost reduction.

1 Introduction

Thanks to their excellent performance for sequence modeling and scalability, transformers [56] have served as the backbone neural network architecture across various important domains, such as natural language processing (NLP) [56, 14, 58, 46, 54], computer vision [16, 37, 5] and speech processing [15, 22, 28]. However, despite their unprecedented popularity, modern transformers suffer from high model complexity, causing expensive costs in both training and inference phases.

To alleviate these challenging issues, **sparse training**, a strategy that originated for optimizing convolutional neural networks (CNNs), has emerged as an attractive solution for efficient transformers. In general, given an input model χ, sparse training aims to output a sparse model χ_s with high task performance. As illustrated in Fig. 1, when χ is randomly initialized, sparse training serves as an efficient *pre-training* method that can reduce the computational and memory costs of the expensive training-from-scratch process. When χ is a pre-trained high-performance dense model, sparse training is essentially the well-known pruning technique, which imposes the sparsity on χ in the *fine-tuning* process and trims down the inference cost.

Although sparse training has been well studied for slimming CNN models, the fundamentally different mechanism of transformers, *e.g.*, multi-head self-attention, make the existing CNN-oriented solutions

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

	Faster Training	Faster Inference	Smaller Model
EcoSpa	✓	✓	✓
GaLore	✗	✗	✗
SLTrain	✗	✗	✓
S²ViTE	✗	✓	✓

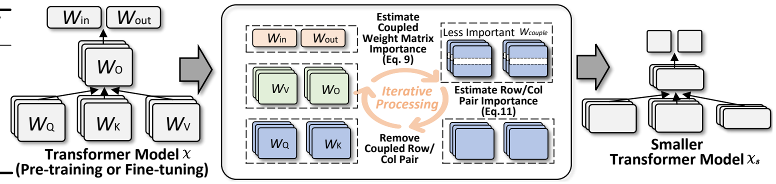

Figure 1: Our proposed EcoSpa is a structured sparse training approach, applicable for both pre-training and fine-tuning stages.

not suitable for transformers. To date, the efficient sparse transformer training is still under-explored. More specifically, 1) for applying sparse training at the pre-training stage, [20], [6, 11] propose to train sparse vision and NLP transformers from scratch, with focus on exploring head-level sparsity. Also [12] proposes to use specially structured matrix for sparse transformer training; 2) for applying sparse training at the fine-tuning stage, [39, 11, 63] explore the structured sparsity at head levels. Observing the potential layer-wise redundancy, [10, 41] propose to remove unimportant layers in the transformer architecture. In addition, [4, 55] use finer granularity at row and column level for structured pruning, achieving good compression performance.

Technical Contributions. In this paper, we propose to perform structured sparse transformer training [1] from a new perspective. Unlike prior works focusing on evaluating the importance of individual components of transformer architecture, *e.g.*, a weight matrix or its component row/column, we propose to fully consider the inter-matrix and inter-row/column correlation, leading to new mechanism for importance assessment and structural removal. More specifically, at the *coupled estimation* step, we introduce a new granularity, namely, coupled weight matrices, to calibrate the architectural importance of transformer model, providing more fine-grained measurement via exploring the inherent coupling effects of weight matrices. Then, at the *coupled sparsification* step, for the row and column within those unimportant coupled matrices, we propose to rank and remove them in a coupled way, maximally preserving the inherent inter-row/column correlation in the weight matrix.

We apply our approach, namely EcoSpa, in both pre-training and fine-tuning scenarios. Evaluation across different model types (*e.g.*, vision transformers, and large language model (LLMs)) show that EcoSpa brings significant performance improvement with respect to training speed, model accuracy, and cost reduction. For instance, when training the GPT-2 model from scratch, our approach can bring $1.6\times$ size reduction with 0.6 lower perplexity (PPL). It also brings $1.6\times$ training speedup compared to the existing sparse pre-training methods. For pre-training sparse LLaMA-1B from scratch, our approach cuts GPU memory usage by half, shortens training time by 21%, and boosts inference throughput by $1.6\times$. For training sparse DeiT from scratch, EcoSpa brings 0.8% accuracy increase over the state-of-the-art solutions. Experiments on pruning LLMs also demonstrate the effectiveness of our approach.

2 Background

2.1 Preliminaries

Notation. We represent tensors using boldface calligraphic script, denoted as $\boldsymbol{\mathcal{X}}$. Matrices and vectors are indicated with boldface capital and lowercase letters, such as $\boldsymbol{X}$ and $\boldsymbol{x}$, respectively. Furthermore, non-boldface letters with indices, *e.g.*, $\mathcal{X}(i_1, \cdots, i_d)$, $X(i,j)$, and $x(i)$, denote the entries for a d-dimensional tensor $\boldsymbol{\mathcal{X}}$, a matrix $\boldsymbol{X}$, and a vector $\boldsymbol{x}$, respectively.

Transformer. For Transformer-based models, the key components include the Multi-Head Attention (MHA) and Feed-Forward Network (FFN). More specifically, the MHA operation is defined as follows:

$$\mathrm{MHA}(\boldsymbol{X}_Q, \boldsymbol{X}_K, \boldsymbol{X}_V) = \mathrm{Concat}(head_1, \ldots, head_h)\boldsymbol{W}^O, \tag{1}$$

[1] In this paper we focus on structured sparsity, which can bring measured speedup on off-the-shelf hardware; while the unstructured sparsity typically cannot. Though some GPUs provide hardware support for semi-structured 2:4 sparsity, this feature is only available for the high-end GPUs such as A100 and up, limiting its practice in many budget-limited resource-limited applications.

where $\boldsymbol{X}_Q, \boldsymbol{X}_K, \boldsymbol{X}_V \in \mathbb{R}^{l \times d_m}$ are the length-l input sequences, d_m is the embedding dimension, and h is the number of attention heads. Here each attention head $head_i$ operates as below:

$$head_i = \text{Attention}(\boldsymbol{X}_Q \boldsymbol{W}_i^Q,\ \boldsymbol{X}_K \boldsymbol{W}_i^K,\ \boldsymbol{X}_V \boldsymbol{W}_i^V) = \phi\left(\frac{\boldsymbol{X}_Q \boldsymbol{W}_i^Q (\boldsymbol{X}_K \boldsymbol{W}_i^K)^\top}{\sqrt{d_h}}\right) \boldsymbol{X}_V \boldsymbol{W}_i^V, \tag{2}$$

where $\phi(\cdot)$ represents the softmax function, $\boldsymbol{W}_i^Q$, $\boldsymbol{W}_i^K, \boldsymbol{W}_i^V \in \mathbb{R}^{d_m \times d_h}$, $\boldsymbol{W}^O \in \mathbb{R}^{d_m \times d_m}$, and $d_m = d_h \times h$. The FFN consists of two fully connected layers with a Gaussian Error Linear Unit (GELU) activation function applied in between. Let $\boldsymbol{X}$ represent the input embeddings, and $\boldsymbol{W}_{in} \in \mathbb{R}^{d_m \times d}, \boldsymbol{W}_{out} \in \mathbb{R}^{d \times d_m}, \boldsymbol{b}_{in}, \boldsymbol{b}_{out}$ denote the weight matrices and bias vectors, respectively. The operation of FFN can be then defined as follows:

$$\text{FFN}(\boldsymbol{X}) = \text{GELU}(\boldsymbol{X}\boldsymbol{W}_{in} + \boldsymbol{b}_{in})\boldsymbol{W}_{out} + \boldsymbol{b}_{out}. \tag{3}$$

2.2 Related Works

Sparse Training for Transformer Pre-training. Applying sparse training at the pre-training stage is very attractive for reducing the high complexity of the costly train-from-scratch process. However, unlike the well-studied sparse CNN pre-training, to date sparse transformer pre-training is still under-explored. [6] dynamically extracts and trains sparse sub-networks, either in unstructured or structured ways, thereby alleviating the training memory bottleneck for Vision Transformers. Inspired by the Lottery Ticket Hypothesis originated in CNN, [11] performs early detection of the structured winning lottery tickets for transformers, improving the pre-training efficiency. In [12], a class of structured matrices, which can closely approximate the dense weight matrices, are proposed for hardware-efficient sparse pre-training, accelerating the overall training process. Notice that the methods developed in [11, 6, 12] can also be applied at the fine-tuning stage for transformer pruning. Another related work is [23], which combines low-rank matrix and unstructured sparse matrix to form the pre-training model.

Sparse Training for LLM Pruning. Due to the high costs of pre-trained LLM models, using sparse training to trim down LLM size is a promising solution for efficient deployment. In general, pruning LLM can be performed in either unstructured or structured way. *Unstructured pruning* [21, 50, 66, 50, 61, 40] focus on removing unimportant individual weights, achieving high compression performance but limited computational efficiency due to the unstructured sparsity pattern. *Structured pruning* [39, 61, 4, 3, 8, 7, 67, 62] aims to sparsify some building components of transformer architecture, *e.g.*, head, row, layer, *etc.*, offering wall-clock time inference speedup. To guide the selection of elements to be removed, typical pruning metrics include magnitude-based [50, 3] and loss-based [39, 55]. Magnitude-based methods use the absolute values of weights, whereas loss-based approaches assess the impact of pruning on model loss, often using the gradient information obtained from Taylor expansion.

Sparse Training for CNN Pre-training/Pruning. Sparse training for CNN, at pre-training and fine-tuning stages, has been well-studied in the literature. For efficient sparse CNN pre-training, [43] explores to prune and grow the same amount of weights periodically and iteratively. [44] proposes to automatically adjust the sparsity levels, achieving good scalability and high computational efficiency. [19] uses the gradient-based criteria to grow the weights with Erdos-Renyi Kernel initialization, and a memory-economic scheme is proposed in [64] for training on the edge devices. Recently, [9] proposes to automatically sparsify the CNN model from scratch, obtaining a compact model without iterative fine-tuning. On the other hand, CNN pruning can be roughly categorized to unstructured pruning (for weights) [24, 25] and structured pruning (for channels/filters) [26, 59, 65, 27, 36, 17, 49, 52, 38]. Considering the importance of structured sparsity for inference speedup, most of the state-of-the-art CNN pruning works focus on structured pruning.

3 Method

Fig. 2 shows the overall procedure of EcoSpa, which consists of two key steps. **(1) Coupled Estimation** (Section 3.1). This step assesses the importance of the building components of transformers, using our proposed new calibration granularity at the coupled weight matrix level. **(2) Coupled Sparsification** (Section 3.2). Once the unimportant coupled weight matrices are identified, EcoSpa further

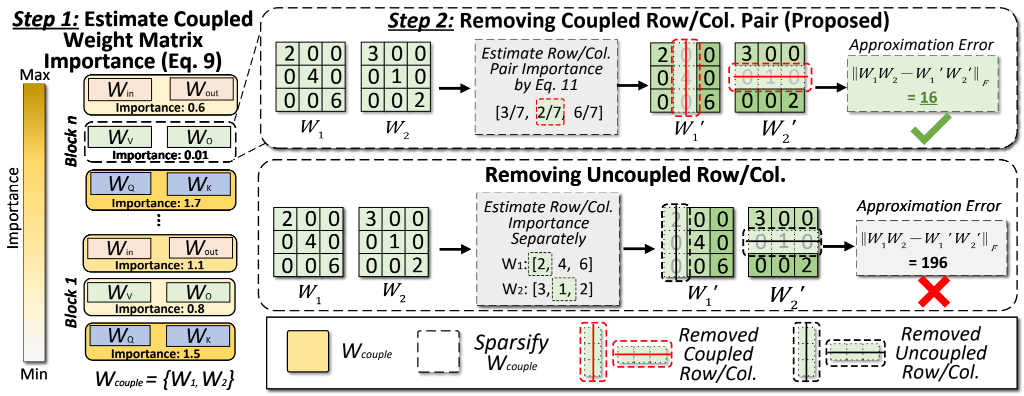

Figure 2: Key steps of EcoSpa: 1) Estimate the importance of coupled weight matrix $\boldsymbol{W}_{\text{couple}}$; and 2) Remove the coupled row/column pair in unimportant $\boldsymbol{W}_{\text{couple}}$.

gauges the fine-grained importance of the coupled row/column pairs in the component matrices and discards the insignificant ones.

3.1 Coupled Estimation: Coupled Weight Matrix-wise Importance Calibration

As extensively studied in the literature [21, 50, 55], estimating the unimportant components of neural networks plays a crucial role for model sparsification. In general, importance estimation can be performed at different granularity levels, such as weight, neuron, layer, and head. More specifically, when aiming for obtaining the structured sparse transformers, identifying the insignificant heads and layers is the most common practice [6, 11, 41, 39].

Unlike existing works, we propose to assess the importance of structural components within transformers using a new granularity, namely *coupled weight matrix*. This idea is motivated by the observation that the transformer has its unique computing pattern and model topology, and hence coupling the weight matrices can provide rich information for importance estimation. Next, we describe the details of our proposal.

Coupled Weight Matrix in MHA. Recall that Eq. 1 and Eq. 2 depict the computations of MHA, which consists of four types of weight matrices $\boldsymbol{W}^Q$, $\boldsymbol{W}^K$, $\boldsymbol{W}^V$ and $\boldsymbol{W}^O$. We propose to estimate the structural importance of MHA by analyzing the combined effect of these matrices. More specifically, consider the following mathematical reformulation of Eq. 2:

$$\text{MHA}(\boldsymbol{X}_Q, \boldsymbol{X}_K, \boldsymbol{X}_V) = \sum_{i=1}^{h} head_i \boldsymbol{W}_i^O = \sum_{i=1}^{h} \phi\Big(\frac{\boldsymbol{X}_Q \overbrace{\boldsymbol{W}_i{}^Q \boldsymbol{W}_i^{K^\top}}^{\boldsymbol{W}_i^{QK}} \boldsymbol{X}_K^\top}{\sqrt{d_h}}\Big) \boldsymbol{X}_V \overbrace{\boldsymbol{W}_i{}^V \boldsymbol{W}_i{}^O}^{\boldsymbol{W}_i^{VO}}, \tag{4}$$

where $\boldsymbol{W}_i^O \in \mathbb{R}^{d_h \times d_m}$, $\boldsymbol{W}^O = \text{Concat}(\boldsymbol{W}_1^O, \ldots, \boldsymbol{W}_h^O)$. It is seen that $\boldsymbol{W}_i^{QK} = \boldsymbol{W}_i^Q \boldsymbol{W}_i^{K^\top}$ and $\boldsymbol{W}_i^{VO} = \boldsymbol{W}_i^V \boldsymbol{W}_i^O$, as the combination of two weight matrices, can serve as the structural components in MHA. Following this perspective, we propose to set the granularity of importance estimation at the level of those coupled weight matrices. We believe this strategy brings two benefits: *i*) it provides more fine-grained measurement than the commonly adopted head-level calibration; and *ii*) it meanwhile naturally explores the inter-matrix correlation within the attention heads, avoiding the limitations if only focusing on individual weight matrices.

Connection to Transformer Circuits Framework. In general, these coupled weight matrices reflect distinct functional roles within attention heads aligning with established transformer circuits framework [18]. Specifically, $\boldsymbol{W}_i^{QK} = \boldsymbol{W}_i^Q \boldsymbol{W}_i^{K^\top}$ governs token-to-token relationships by determining attention patterns, while $\boldsymbol{W}_i^{VO} = \boldsymbol{W}_i^V \boldsymbol{W}_i^O$ captures token influence on the output by modulating and integrating attended information [18]. This coupled structure encapsulates both the selection of relevant information and its transformation into meaningful outputs. A concrete example of this mechanism is observed in induction heads [45], which enable in-context learning by recognizing and extending patterns in sequences. Given an input like "A B … A", the $\boldsymbol{W}_i^{QK}$ matrix facilitates retrieval by computing attention scores to determine the relevance of the current "A" to previous tokens. Meanwhile, $\boldsymbol{W}_i^{VO}$ ensures that the model correctly predicts "B" as the next token by reinforcing the learned sequence. This mechanism is fundamental in applications such as code completion and structured text generation, where leveraging previously seen sequences enhances model consistency and coherence. By preserving these functional circuits through our coupled matrix approach, EcoSpa maintains the essential computational patterns that make transformers effective while achieving efficient sparsification.

Extension to GQA. Next we show that the concept of coupled weight matrix can also be applied to Grouped Query Attention (GQA) [1] – the generalization of MHA that has been adopted in state-of-the-art LLMs such as LLaMA [54], Falcon [2], Mistral [32]. Recall that GQA divides h attention heads into multiple groups, so its attention mechanism can be reformulated as follows:

$$\mathrm{GQA}(\boldsymbol{X}_Q, \boldsymbol{X}_K, \boldsymbol{X}_V) = \sum_{i=1}^{h} \phi\Big(\frac{\boldsymbol{X}_Q \overbrace{\boldsymbol{W}_i^Q \boldsymbol{W}_{g(i)}^{K^\top}}^{\boldsymbol{W}_{i,g(i)}^{QK}} \boldsymbol{X}_K^\top}{\sqrt{d_h}}\Big) \boldsymbol{X}_V \overbrace{\boldsymbol{W}_{g(i)}^V \boldsymbol{W}_i^O}^{\boldsymbol{W}_{g(i),i}^{VO}}, \tag{5}$$

where $g(i)$ maps the i-th head to its corresponding group. It is seen the structural components in the format of coupled weight matrices also exist for GQA, as $\boldsymbol{W}_{i,g(i)}^{QK} = \boldsymbol{W}_i^Q \boldsymbol{W}_{g(i)}^{K^\top}$ and $\boldsymbol{W}_{g(i),i}^{VO} = \boldsymbol{W}_{g(i)}^V \boldsymbol{W}_i^O$.

Compatibility with RoPE. Furthermore, when rotary position embedding (RoPE) [48] is used in the model architecture, the coupled weight matrix can also be incorporated into this position-aware scenario, capturing both the structural coupling and positional encoding as follows:

$$\begin{aligned}\mathrm{MHA}(\boldsymbol{X}_Q, \boldsymbol{X}_K, \boldsymbol{X}_V)_{\mathrm{RoPE}} &= \sum_{i=1}^{h} \phi\Big(\frac{\mathrm{RoPE}(\boldsymbol{X}_Q \boldsymbol{W}_i^Q)\mathrm{RoPE}(\boldsymbol{X}_K \boldsymbol{W}_i^K)^\top}{\sqrt{d_h}}\Big) \boldsymbol{X}_V \boldsymbol{W}_i^{VO} \\ &= \sum_{i=1}^{h} \phi\Big(\frac{\boldsymbol{X}_Q \boldsymbol{W}_i^Q \boldsymbol{P}_t \boldsymbol{P}_s^\top \boldsymbol{W}_i^{K^\top} \boldsymbol{X}_K^\top}{\sqrt{d_h}}\Big) \boldsymbol{X}_V \boldsymbol{W}_i^{VO} = \sum_{i=1}^{h} \phi\Big(\frac{\boldsymbol{X}_Q \overbrace{\boldsymbol{W}_{i,\mathrm{RoPE}}^Q \boldsymbol{W}_{i,\mathrm{RoPE}}^{K^\top}}^{\boldsymbol{W}_{i,RoPE}^{QK}} \boldsymbol{X}_K^\top}{\sqrt{d_h}}\Big) \boldsymbol{X}_V \boldsymbol{W}_i^{VO},\end{aligned} \tag{6}$$

where $\boldsymbol{W}_{i,\mathrm{RoPE}}^Q = \boldsymbol{W}_i^Q \boldsymbol{P}_t$ and $\boldsymbol{W}_{i,\mathrm{RoPE}}^K = \boldsymbol{W}_i^K \boldsymbol{P}_s$. Here $\mathrm{RoPE}(\cdot)$ encodes positional information by applying position-dependent rotations to query and key projections, where $\boldsymbol{P}_t, \boldsymbol{P}_s \in \mathbb{R}^{d_h \times d_h}$ are rotation matrices corresponding to the t-th and s-th positions, respectively. Each matrix is block-diagonal, with blocks defined as: $\begin{pmatrix} \cos(t\theta_j) & \sin(t\theta_j) \\ -\sin(t\theta_j) & \cos(t\theta_j) \end{pmatrix}, \begin{pmatrix} \cos(s\theta_j) & \sin(s\theta_j) \\ -\sin(s\theta_j) & \cos(s\theta_j) \end{pmatrix}$, where $j \in \{1 \cdots d_h/2\}$ and θ_j is pre-defined frequency parameter, *e.g.*, $\theta_j = 1/10000^{2j/d_h}$. From Eq. 6 it is seen that with RoPE integrated, the positional encoding is naturally incorporated into the weight transformations, redefining the coupled weight matrix as $\boldsymbol{W}_{i,RoPE}^{QK} = \boldsymbol{W}_{i,\mathrm{RoPE}}^Q \boldsymbol{W}_{i,\mathrm{RoPE}}^{K^\top}$.

Coupling Effect of Weight Matrices in FFN. Following the same philosophy, we also evaluate the importance of the building blocks in FFN from the lens of the coupled weight matrix. More specifically, consider the FFN computation (Eq. 3) can be reformulated as follows:

$$\begin{aligned}\mathrm{FFN}(\boldsymbol{X}) &= 0.5\boldsymbol{Y}_1 \odot (1 + \mathrm{erf}(\boldsymbol{Y}_1/\sqrt{2})\boldsymbol{W}_{\mathrm{out}} \\ &= 0.5\boldsymbol{X}\boldsymbol{W}_{\mathrm{in}}\boldsymbol{W}_{\mathrm{out}} + 0.5\mathrm{erf}(\boldsymbol{Y}_1/\sqrt{2}) \odot (\boldsymbol{X}\boldsymbol{W}_{\mathrm{in}})\boldsymbol{W}_{\mathrm{out}},\end{aligned} \tag{7}$$

where $\boldsymbol{Y}_1 = \boldsymbol{X}\boldsymbol{W}_{in}$, $\odot$ is the Hadamard product, and $\mathrm{erf}(\cdot)$ denotes the error function as $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-t^2)dt$ bounded by $[-1, 1]$. This formulation highlights that the coupled weight matrix $\boldsymbol{W}_{\mathrm{io}} = \boldsymbol{W}_{\mathrm{in}}\boldsymbol{W}_{\mathrm{out}}$ in the first term of Eq. 7 forms the backbone of FFN computations. While the second correction term modulated by $\mathrm{erf}(\cdot)$ introduces variability, it is always not larger than the first term, since it is bounded by $[-0.5\boldsymbol{X}\boldsymbol{W}_{\mathrm{in}}\boldsymbol{W}_{\mathrm{out}}, 0.5\boldsymbol{X}\boldsymbol{W}_{\mathrm{in}}\boldsymbol{W}_{\mathrm{out}}]$. Based on this observation, we propose to approximately assess the structural importance of FFN via evaluating the importance of $\boldsymbol{W}_{\mathrm{in}}\boldsymbol{W}_{\mathrm{out}}$.

Coupled Weight Matrix-wise Importance. After setting the resolution of the importance estimator with the proposed granularity, we can gauge the structural criticality of MHA and FFN accordingly. To that end, we calibrate the empirical Fisher information [34] in a coupled weight matrix-wise way as follows:

$$\hat{I}(\boldsymbol{W}_{\text{couple}}) = \frac{1}{2}\sum_{k=1}^{2}\frac{1}{|\boldsymbol{W}_k|}\frac{1}{|\boldsymbol{\mathcal{D}}|}\sum_{i,j}\Big(\frac{\partial\mathcal{L}(\boldsymbol{W}_k;\boldsymbol{\mathcal{D}})}{\partial\boldsymbol{W}_k}\Big)^2_{i,j}, \tag{8}$$

where $\boldsymbol{W}_{\text{couple}} = \boldsymbol{W}_1\boldsymbol{W}_2$. Here for MHA, $\{\boldsymbol{W}_1, \boldsymbol{W}_2\}$ is $\{\boldsymbol{W}_i^Q, \boldsymbol{W}_i^{K^\top}\}$ and $\{\boldsymbol{W}_i^V, \boldsymbol{W}_i^O\}$; while for FFN, $\{\boldsymbol{W}_1, \boldsymbol{W}_2\}$ is $\{\boldsymbol{W}_{\text{in}}, \boldsymbol{W}_{\text{out}}\}$ [2]. Also, $\boldsymbol{\mathcal{D}}$ is the training data and $|\cdot|$ returns the size of the operand. Notice that different from prior works [29, 51], the empirical Fisher information is calculated for the entire coupled weight matrix $\boldsymbol{W}_{\text{couple}}$ instead of its entries. Smaller $\hat{I}(\boldsymbol{W}_{\text{couple}})$ indicates the lower importance of $\boldsymbol{W}_{\text{couple}}$ when sparsifying the model.

3.2 Coupled Sparsification: Removing the Coupled Row/Column Pair

Upon obtaining the importance information of all the coupled weight matrices, the next step is to sparsify $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ belonging to those less significant $\boldsymbol{W}_{\text{couple}}$. To that end, we perform row/column-wise sparsification, a strategy with finer granularity than removing the entire $\boldsymbol{W}_i$, towards minimizing performance loss and preserving model structuredness. Notice that though removing the rows/columns in the weight matrices of transformers has been reported in [11, 6, 55], we believe those individual weight matrix-oriented methods are not the best suited in the scenario involved with coupled weight matrices. In other words, the coupling effect between $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ should be fully considered and leveraged in the sparsification process. This principle is supported by our theoretical analysis in Appendix A and B. Aiming at that, we propose coupled sparsification, a solution that removes the coupled row/column pairs in $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$, with details described as below.

Inspiration from tSVD. The key idea of coupled sparsification is inspired by the philosophy of truncated singular value decomposition (tSVD). More specifically, recall that a matrix $\boldsymbol{M} \in \mathbb{R}^{m\times n}$ can be exactly factorized to two matrices via SVD as:

$$\begin{aligned}\boldsymbol{M} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top &= \begin{pmatrix}\boldsymbol{U}_{m,r} \\ \boldsymbol{U}_{m,(m-r)}\end{pmatrix}^\top \begin{pmatrix}\boldsymbol{\Sigma}_{r,r} & \\ & \boldsymbol{\Sigma}_{(m-r),(n-r)}\end{pmatrix}\begin{pmatrix}\boldsymbol{V}_{r,n}^\top \\ \boldsymbol{V}_{(n-r),n}^\top\end{pmatrix} \\ &= \begin{pmatrix}\boldsymbol{U}_{m,r}\boldsymbol{\Sigma}_{r,r}^{1/2} \\ \boldsymbol{U}_{m,(m-r)}\boldsymbol{\Sigma}_{(m-r),(n-r)}^{1/2}\end{pmatrix}^\top \begin{pmatrix}\boldsymbol{\Sigma}_{r,r}^{1/2}V_{r,n}^\top \\ \boldsymbol{\Sigma}_{(m-r),(n-r)}^{1/2}V_{(n-r),n}^\top\end{pmatrix} = \boldsymbol{M}_1\boldsymbol{M}_2,\end{aligned} \tag{9}$$

where $\boldsymbol{\Sigma}_{r,r} \in \mathbb{R}^{r\times r}$ is a diagonal matrix containing r largest singular values σ_i's and $\boldsymbol{\Sigma}_{(m-r),(n-r)} \in \mathbb{R}^{(m-r)\times(n-r)}$ is a diagonal rectangular matrix containing the smallest $(\min(m,n) - r)$ σ_i's. Then, consider a rank-r tSVD [57] for approximating $\boldsymbol{M}$ is performed as:

$$\boldsymbol{M} \approx (\boldsymbol{U}_{m,r}\boldsymbol{\Sigma}_{r,r}^{1/2})(\boldsymbol{\Sigma}_{r,r}^{1/2}\boldsymbol{V}_{r,n}^\top) = (\boldsymbol{M}_1 \odot \boldsymbol{S}_{\text{mask1}})(\boldsymbol{M}_2 \odot \boldsymbol{S}_{\text{mask2}}). \tag{10}$$

Notice that here $\boldsymbol{S}_{\text{mask}i}$ is the binary matrix that essentially removes a sets of rows/columns of $\boldsymbol{M}_i$. Meanwhile, it is theoretically proven that tSVD provides the *optimal* rank-r approximation for $\boldsymbol{M}$ [57]. Therefore, the row/column-wise sparsification policy for $\boldsymbol{M}_1$ and $\boldsymbol{M}_2$ in tSVD, by its nature, brings important insights for sparsifying two component matrices with closely approximating their combination. More specifically, comparing Eq. 9 and Eq. 10, we can see that the removed rows/columns are $\boldsymbol{U}_{m,(m-r)}\boldsymbol{\Sigma}_{(m-r),(n-r)}^{1/2}$ and $\boldsymbol{\Sigma}_{(m-r),(n-r)}^{1/2}\boldsymbol{V}_{(n-r),n}^\top$, which exhibit the following characteristics:

<u>**Observation #1**</u> (Small ℓ_2 Norm) *The removed rows/columns of $\boldsymbol{M}_1$ and $\boldsymbol{M}_2$ typically have smaller ℓ_2 norm, as $\boldsymbol{\Sigma}_{(m-r),(n-r)}$ only contains very least significant σ_i's.*

<u>**Observation #2**</u> (Aligned Removal) *The removed rows in $\boldsymbol{M}_1$ and columns in $\boldsymbol{M}_2$ always have the same indices, exhibiting the positional alignment of the sparsification.*

Sparsifying Coupled Weight Matrices. Considering the mathematical format of Eq. 10 is highly similar to our desired task of sparsifying the coupled $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$, a natural idea is directly performing tSVD (Eq. 10) on $\boldsymbol{W} = \boldsymbol{W}_1\boldsymbol{W}_2$ to obtain sparse $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$. However, this strategy is

[2] FFN in LLaMA models contains three weight matrices ($\boldsymbol{W}_{\text{gate}}$, $\boldsymbol{W}_{\text{up}}$, and $\boldsymbol{W}_{\text{down}}$) and SwiGLU. The coupled structure in this scenario is established by defining $\boldsymbol{W}_1 = \boldsymbol{W}_{\text{gate}} \odot \boldsymbol{W}_{\text{up}}$, $\boldsymbol{W}_2 = \boldsymbol{W}_{\text{down}}$, preserving the key relationships between the weight matrices.

mathematically infeasible because Eq. 9 is generally not invertible, *e.g.*, we cannot use $\mathrm{SVD}(\boldsymbol{W}_1\boldsymbol{W}_2)$ to reconstruct $\boldsymbol{W}_1$ or $\boldsymbol{W}_2$. Fortunately, we can still leverage the observations extracted from tSVD process to design the sparsification policy for $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$. More specifically, we propose to calculate the importance scores of the row/column vectors of $\boldsymbol{W}_i$:

$$\boldsymbol{v} = [v_1, v_2, \dots, v_d] = \frac{\|\mathrm{vec}(\boldsymbol{W}_1)\|_{2,\mathrm{col}} \odot \|\mathrm{vec}(\boldsymbol{W}_2)\|_{2,\mathrm{row}}}{\|\|\mathrm{vec}(\boldsymbol{W}_1)\|_{2,\mathrm{col}} \odot \|\mathrm{vec}(\boldsymbol{W}_2)\|_{2,\mathrm{row}}\|_2}, \tag{11}$$

where $\|\mathrm{vec}(\cdot)\|_{2,\mathrm{col}}$ and $\|\mathrm{vec}(\cdot)\|_{2,\mathrm{row}}$ represent the column-wise and row-wise ℓ_2 norm of a matrix, respectively, *e.g.*, $\|\mathrm{vec}(\boldsymbol{W}_1)\|_{2,\mathrm{col}} = \sqrt{\sum_{i=1}^{m} |\boldsymbol{W}_1(i,j)|}, (j = 1, 2, \dots, d)\}$. And $\|\mathrm{vec}(\cdot)\|_2$ calculates the ℓ_2 norm of a vector. From Eq. 11 it is seen that v_i essentially calculates the normalized combined ℓ_2 for the i-th column of $\boldsymbol{W}_1$ and the i-th row of $\boldsymbol{W}_2$, reflecting the two key observations we obtain from tSVD. Hence we can rank v_i's to determine the important coupled row/column in $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$, and then remove those insignificant pairs. Note that when RoPE is used in the attention, $\boldsymbol{P}_t$ and $\boldsymbol{P}_s^\top$ are incorporated into the process of calculating importance scores for the row and column pairs of $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$, while the sparsification process is still performed on the individual weight matrices such as $\boldsymbol{W}_i^Q$ and $\boldsymbol{W}_i^K$. $\boldsymbol{P}_t$ and $\boldsymbol{P}_s^\top$ will be then adjusted according to the new shapes of weight matrices.

Overall Sparse Training Procedure.

Algorithm 1 describes the processing scheme of EcoSpa. Here in each epoch, after identifying top-K least significant W_{couple}, the importance scores of the rows/columns in those component weight matrices are calculated. Once the cumulative importance scores exceed the threshold θ, the pair of rows and columns corresponding to the smallest score is removed. This gradual sparsification process continues till the overall model reaches the target budget size. We analyze and discuss the selection of hyper-parameters in Appendix C.

Algorithm 1: Processing Scheme of EcoSpa

```
Input: Random Initialized/Pre-trained model 𝓦,
       top-K ratio, Target model size c,
       Cumulative threshold θ
Output: Sparse Model 𝓦̂
for t in [1, 2, ··· , T] do
    Update: 𝓦_t = 𝓦_{t−1} + δ𝓦
    if Param(𝓦_t) > c then
        Compute: Î(W_couple), ∀W_couple ∈ 𝓦                ▷ Eq. 8
        Select: top-K W_couple with smallest
          Î → {W_couple}^{top−K}
        for W_couple in {W_couple}^{top−K} do
            Compute: v of W_couple                        ▷ Eq. 11
            while Σ v_i > θ do
                Remove smallest v_i and i-th row/col. of
                  W_couple
                Remove i-th row/col. optimizer moments of
                  W_couple
```

4 Experiments

4.1 Experiments on Pre-training (Train from Scratch)

Large Language Model (LLaMA): For large language models, we follow the training settings in [69] to train sparse LLaMA from scratch on the C4 dataset. The training employs the Galore optimizer with an initial learning rate of 0.01 and a batch size of 512.

NLP Transformer (GPT-2): We pre-train sparse NLP transformers by following the training settings in [68]. Specifically, we train sparse GPT-2 [47] models from scratch on the WikiText-103 dataset [42], using the AdamW optimizer with an initial learning rate of 0.001, training for 100 epochs and a batch size of 512.

Vision Transformer (DeiT): We apply our approach to train sparse DeiT models [53] from scratch on the ImageNet-1K dataset [13]. We use the same hyper-parameters as in [6], which include the AdamW optimizer with an initial learning rate of 0.0005 and a batch size of 512.

Table 1: Evaluation of training LLaMA-1B models from scratch on C4 dataset on 13.1B tokens using 8×A100 GPUs. Validation perplexity is provided, along with memory estimates for total parameters and optimizer states in BF16 format. The results for LoRA and ReLoRA are sourced from [69].

Method	PPL	Training Time	# Params. (M)	# Mem. (GB)	Throughput (tokens/s)
Baseline	15.56	51.1h	1339.08	7.80	21786.49
LoRA	19.21	125.4h	1339.08	6.17	21786.49
ReLoRA	18.33	125.6h	1339.08	6.17	21786.49
Galore	15.64	60.6h	1339.08	4.38	21786.49
EcoSpa	15.60	**40.3h**	**933.94**	**3.88**	**35211.27**

Table 2: Evaluation of training LLaMA-7B models from scratch on the C4 dataset (1.4B tokens) using 8×A100 GPUs.

Method	PPL	Training Time	# Params. (M)	# Mem. (GB)	Throughput (tokens/s)
8-bit Galore	26.9	20.8h	6738.42	17.86	9689.9
8-bit SLtrain	27.6	31.4h	3144.73	11.81	2204.6
8-bit EcoSpa	23.0	**18.3h**	5046.96	12.56	**10330.6**

Table 3: Evaluation of training sparse/low-rank GPT-2 models from scratch on WikiText-103 using 8×A100 GPUs. Validation perplexity is provided, along with memory estimates for total parameters and optimizer states in FP16 format.

Method	PPL	Training Time	# Params. (M)	# Mem. (MB)	Throughput (tokens/s)
GPT2-Small	18.5	9.5h	124.4	711.8	47411.2
In-Rank	18.9	-	91.2	521.9	-
Monarch	20.7	-	72	412.0	-
EcoSpa	**17.8**	**7.5h**	**71.2**	**504.7**	**80691.2**
GPT2-Medium	19.5	28.7h	354.8	2030.2	36556.8
In-Rank	20.2	18.0h	223.0	1276.0	-
Monarch	20.3	-	165	994.1	-
EcoSpa	**17.1**	**17.3h**	**158.5**	**917.2**	**52326.4**

Comparison Results. Table 1 presents the pre-training results for LLaMA-1B. Our approach reduces GPU memory usage by 50%, decreases training time by 21%, and achieves a $1.6\times$ speedup in inference throughput without any performance loss. Compared to Galore, our approach reduces memory usage by 10%, increases training speed by $1.5\times$, and boosts inference speed by $1.6\times$. The results for LoRA [30] and ReLoRA [35] are sourced from [69]. For low-rank methods, LoRA [31] fine-tunes pre-trained models using low-rank adaptors: $W = W_0 + BA$, where W_0 is the fixed initial weights and BA is a learnable low-rank adaptor. For pre-training, W_0 is the full-rank initialization matrix. ReLoRA [35] is a variant of LoRA designed for pre-training. It periodically merges BA into W and reinitializes BA with a reset on optimizer states and learning rate. EcoSpa surpasses these low-rank methods, reducing PPL by 3.6 and 2.7, respectively, while decreasing memory usage by 37% and offering $1.6\times$ acceleration in inference throughput.

Table 2 presents the results of pre-training LLaMA-7B. Compared to SLTrain [23], EcoSpa reduces training time by 41.7% and achieves a $4.7\times$ speedup in inference with better performance. Unlike SLTrain adopting the combination of low-rank factorization and unstructured sparsity that cannot translate to actual speedup, the training process of EcoSpa is on the structured sparse models that enjoy measured training and inference speedup.

Table 3 compares our approach with the existing sparse and low-rank pre-training works. It is seen that EcoSpa achieves better performance than baseline and prior efforts using less training time. Specifically, our approach can train sparse GPT2-Small and GPT2-Medium models from scratch, with $1.7\times$ and $2.2\times$ model size reduction, respectively; and meanwhile, it brings 0.7 and 2.4 lower perplexity over the baseline models.

Table 4 lists the performance results of various sparse vision transformer pre-training methods. EcoSpa achieves a 0.8% increase in top-1 accuracy for DeiT-Tiny and a 0.35% increase for DeiT-Small over state-of-the-art solutions, along with greater model size reduction.

Table 4: Results of training sparse DeiT models from scratch on ImageNet-1k. Results for SSP-Tiny and SSP-Small are sourced from [6]. Inference speedup is measured on Nvidia RTX 3090 GPU.

Method	# Params. (M)	# Mem. (MB)	FLOPs Saving (%)	Top-1 Acc. (%)	Inference Speedup
DeiT-Tiny	5.72	32.73	-	71.80	-
SSP-Tiny	4.21	24.09	23.69	68.59	1.12×
S^2ViTE-Tiny	4.21	24.09	23.69	70.12	1.12×
EcoSpa	**3.95**	**22.60**	**32.01**	70.92	**1.20**×
DeiT-Small	22.10	126.46	-	79.78	-
SSP-Small	14.60	83.54	33.13	77.74	1.29×
S^2ViTE-Small	14.60	83.54	33.13	79.22	1.29×
EcoSpa	**13.98**	**79.99**	**37.30**	79.57	**1.29**×

Table 5: Perplexity of compressed LLaMA2-7B on WikiText-2 with different target model sizes. SVD-LLM [60] and SliceGPT [4] are low-rank based methods. LLM Surgeon [55] is a pruning method, K-OBD [55], as a baseline comparison method, uses Kronecker-factored curvature and only prunes without updating the remaining weights.

Method		K-OBD	SVD-LLM	LLM Surgeon	SliceGPT	**EcoSpa**
Training Time		16h58m, H100	15m, A100	17h08m, H100	1h07m, H100	**1h41m, A100**
PPL @ Target Size	80%	9.14	7.94	6.18	6.64	6.36
	70%	15.43	9.56	7.83	8.12	**7.66**
	60%	28.03	13.11	10.39	-	**10.24**
	50%	46.64	23.97	15.38	-	**14.02**

Table 6: Comparison of downstream zero-shot task performance of LLaMA2-7B model when trained on WikiText2 dataset.

	Target Size	PIQA	WinoGrande	HellaSwag	ARC-e	ARC-c	Avg.
LLaMA2-7B	100%	79.11	69.06	75.99	74.58	46.25	69.00
EcoSpa	80%	73.78	61.48	67.79	61.62	39.42	60.82
	75%	71.93	60.69	64.69	54.38	35.15	57.37
	70%	70.18	59.98	60.00	49.16	34.22	54.71
SliceGPT	80%	69.42	65.11	59.04	59.76	37.54	58.18
	75%	66.87	63.38	54.16	58.46	34.56	55.48
	70%	63.55	61.33	49.62	51.77	31.23	51.50

4.2 Experiments on Fine-tuning LLaMA2-7B (Structured Pruning)

Experimental Setting. We evaluate the pruning performance of EcoSpa on the pre-trained LLaMA2 [54] models. We use WikiText-2 [42] as the calibration dataset and evaluate the perplexity of the pruned model. We follow the same training settings adopted in [55], use 128 sequences with a sequence length of 2048 tokens from the training dataset, and evaluate perplexity on the standard test split. Additionally, we also evaluate the performance of the pruned models on downstream zero-shot tasks.

Comparison Results. Table 5 presents a comparison of the perplexity performance between EcoSpa and existing LLM pruning and low-rank factorization methods applied to LLaMA2-7B. Our approach consistently achieves lower perplexity across various target model size configurations compared to previous works. Additionally, Table 6 illustrates the zero-shot task performance of the pruned LLaMA2-7B model. In comparison to SliceGPT [4], our method demonstrates improved results across different target model sizes, indicating its effectiveness.

5 Conclusion

In this paper, we propose EcoSpa, an efficient structured sparse training approach for transformers. By estimating the coupled weight matrix-wise importance and removing the coupled row/column pair during training, EcoSpa brings a significant reduction in training costs and model complexity with preserving high task performance. Experiments across various transformer models demonstrate the superior performance of EcoSpa in both pre-training and fine-tuning scenarios.

References

[1] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

[2] Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, et al. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*, 2023.

[3] Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. Fluctuation-based adaptive structured pruning for large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 10865–10873, 2024.

[4] Saleh Ashkboos, Maximilian L Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. Slicegpt: Compress large language models by deleting rows and columns. *arXiv preprint arXiv:2401.15024*, 2024.

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[6] Tianlong Chen, Yu Cheng, Zhe Gan, Lu Yuan, Lei Zhang, and Zhangyang Wang. Chasing sparsity in vision transformers: An end-to-end exploration. *Advances in Neural Information Processing Systems*, 34:19974–19988, 2021.

[7] Tianyi Chen, Tianyu Ding, Badal Yadav, Ilya Zharkov, and Luming Liang. Lorashear: Efficient large language model structured pruning and knowledge recovery. *arXiv preprint arXiv:2310.18356*, 2023.

[8] Tianyi Chen, Luming Liang, Tianyu DING, Zhihui Zhu, and Ilya Zharkov. OTOv2: Automatic, generic, user-friendly. In *The Eleventh International Conference on Learning Representations*, 2023.

[9] Tianyi Chen, Luming Liang, Tianyu Ding, Zhihui Zhu, and Ilya Zharkov. Otov2: Automatic, generic, user-friendly. *arXiv preprint arXiv:2303.06862*, 2023.

[10] Xiaodong Chen, Yuxuan Hu, and Jing Zhang. Compressing large language models by streamlining the unimportant layer. *arXiv preprint arXiv:2403.19135*, 2024.

[11] Xiaohan Chen, Yu Cheng, Shuohang Wang, Zhe Gan, Zhangyang Wang, and Jingjing Liu. Earlybert: Efficient bert training via early-bird lottery tickets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2195–2207, 2021.

[12] Tri Dao, Beidi Chen, Nimit S Sohoni, Arjun Desai, Michael Poli, Jessica Grogan, Alexander Liu, Aniruddh Rao, Atri Rudra, and Christopher Ré. Monarch: Expressive structured matrices for efficient and accurate training. In *International Conference on Machine Learning*, pages 4690–4721. PMLR, 2022.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

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

[15] Linhao Dong, Shuang Xu, and Bo Xu. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5884–5888. IEEE, 2018.

[16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[17] Abhimanyu Dubey, Moitreya Chatterjee, and Narendra Ahuja. Coreset-based neural network compression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 454–470, 2018.

[18] Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12, 2021.

[19] Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *International Conference on Machine Learning*, pages 2943–2952. PMLR, 2020.

[20] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[21] Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR, 2023.

[22] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. Conformer: Convolution-augmented transformer for speech recognition. *arXiv preprint arXiv:2005.08100*, 2020.

[23] Andi Han, Jiaxiang Li, Wei Huang, Mingyi Hong, Akiko Takeda, Pratik Jawanpuria, and Bamdev Mishra. Sltrain: a sparse plus low-rank approach for parameter and memory efficient pretraining. *arXiv preprint arXiv:2406.02214*, 2024.

[24] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[25] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

[26] Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4340–4349, 2019.

[27] Zejiang Hou, Minghai Qin, Fei Sun, Xiaolong Ma, Kun Yuan, Yi Xu, Yen-Kuang Chen, Rong Jin, Yuan Xie, and Sun-Yuan Kung. Chex: Channel exploration for cnn model compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12287–12298, 2022.

[28] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

[29] Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. Language model compression with weighted low-rank factorization. *arXiv preprint arXiv:2207.00112*, 2022.

[30] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

[31] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2021.

[32] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[33] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[34] Frederik Kunstner, Philipp Hennig, and Lukas Balles. Limitations of the empirical fisher approximation for natural gradient descent. *Advances in neural information processing systems*, 32, 2019.

[35] Vladislav Lialin, Sherin Muckatira, Namrata Shivagunde, and Anna Rumshisky. Relora: High-rank training through low-rank updates. In *The Twelfth International Conference on Learning Representations*, 2023.

[36] Mingbao Lin, Rongrong Ji, Yan Wang, Yichen Zhang, Baochang Zhang, Yonghong Tian, and Ling Shao. Hrank: Filter pruning using high-rank feature map. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1529–1538, 2020.

[37] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[38] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, pages 5058–5066, 2017.

[39] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720, 2023.

[40] Srikanth Malla, Joon Hee Choi, and Chiho Choi. Copal: Continual pruning in large language generative models. *arXiv preprint arXiv:2405.02347*, 2024.

[41] Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. Shortgpt: Layers in large language models are more redundant than you expect. *arXiv preprint arXiv:2403.03853*, 2024.

[42] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

[43] Decebal Constantin Mocanu, Elena Mocanu, Peter Stone, Phuong H Nguyen, Madeleine Gibescu, and Antonio Liotta. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature communications*, 9(1):2383, 2018.

[44] Hesham Mostafa and Xin Wang. Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. In *International Conference on Machine Learning*, pages 4646–4655. PMLR, 2019.

[45] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

[46] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, and et al. Gpt-4 technical report, 2023.

[47] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[48] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[49] Yang Sui, Miao Yin, Yi Xie, Huy Phan, Saman Aliari Zonouz, and Bo Yuan. Chip: Channel independence-based pruning for compact neural networks. *Advances in Neural Information Processing Systems*, 34:24604–24616, 2021.

[50] Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*, 2023.

[51] Yi-Lin Sung, Varun Nair, and Colin A Raffel. Training neural networks with fixed sparse masks. *Advances in Neural Information Processing Systems*, 34:24193–24205, 2021.

[52] Chong Min John Tan and Mehul Motani. Dropnet: Reducing neural network complexity via iterative pruning. In *International Conference on Machine Learning*, pages 9356–9366. PMLR, 2020.

[53] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, volume 139, pages 10347–10357, July 2021.

[54] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[55] Tycho FA van der Ouderaa, Markus Nagel, Mart Van Baalen, Yuki M Asano, and Tijmen Blankevoort. The llm surgeon. *arXiv preprint arXiv:2312.17244*, 2023.

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[57] Michael E Wall, Andreas Rechtsteiner, and Luis M Rocha. Singular value decomposition and principal component analysis. In *A practical approach to microarray data analysis*, pages 91–109. Springer, 2003.

[58] Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao. Learning deep transformer models for machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1810–1822, 2019.

[59] Wenxiao Wang, Cong Fu, Jishun Guo, Deng Cai, and Xiaofei He. Cop: Customized deep model compression via regularized correlation-based filter-level pruning. *arXiv preprint arXiv:1906.10337*, 2019.

[60] Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*, 2024.

[61] Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared llama: Accelerating language model pre-training via structured pruning. *arXiv preprint arXiv:2310.06694*, 2023.

[62] Yifei Yang, Zouying Cao, and Hai Zhao. Laco: Large language model pruning via layer collapse. *arXiv preprint arXiv:2402.11187*, 2024.

[63] Shixing Yu, Tianlong Chen, Jiayi Shen, Huan Yuan, Jianchao Tan, Sen Yang, Ji Liu, and Zhangyang Wang. Unified visual transformer compression. *arXiv preprint arXiv:2203.08243*, 2022.

[64] Geng Yuan, Xiaolong Ma, Wei Niu, Zhengang Li, Zhenglun Kong, Ning Liu, Yifan Gong, Zheng Zhan, Chaoyang He, Qing Jin, et al. Mest: Accurate and fast memory-economic sparse training framework on the edge. *Advances in Neural Information Processing Systems*, 34:20838–20850, 2021.

[65] Edouard Yvinec, Arnaud Dapogny, Matthieu Cord, and Kevin Bailly. Red: Looking for redundancies for data-freestructured compression of deep neural networks. *Advances in Neural Information Processing Systems*, 34:20863–20873, 2021.

[66] Yuxin Zhang, Lirui Zhao, Mingbao Lin, Yunyun Sun, Yiwu Yao, Xingjia Han, Jared Tanner, Shiwei Liu, and Rongrong Ji. Dynamic sparse no training: Training-free fine-tuning for sparse llms. *arXiv preprint arXiv:2310.08915*, 2023.

[67] Bowen Zhao, Hannaneh Hajishirzi, and Qingqing Cao. Apt: Adaptive pruning and tuning pretrained language models for efficient training and inference. *arXiv preprint arXiv:2401.12200*, 2024.

[68] Jiawei Zhao, Yifei Zhang, Beidi Chen, Florian Schäfer, and Anima Anandkumar. Inrank: Incremental low-rank learning. 2023.

[69] Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient llm training by gradient low-rank projection. *arXiv preprint arXiv:2403.03507*, 2024.

Appendix

A Theoretical Analysis

A.1 Problem Setup

Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ be a matrix whose columns are $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n \in \mathbb{R}^m$. Let $\boldsymbol{B} \in \mathbb{R}^{n \times p}$ be a matrix whose rows are $\boldsymbol{b}_1^T, \ldots, \boldsymbol{b}_n^T$, where $\boldsymbol{b}_i \in \mathbb{R}^p$. The product $\boldsymbol{AB}$ can be expressed as a sum of outer products:

$$\boldsymbol{P} \coloneqq \boldsymbol{AB} = \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{b}_i^T \tag{12}$$

We analyze the error introduced by removing a single column from $\boldsymbol{A}$ and a single row from $\boldsymbol{B}$. Let $\boldsymbol{A}_{\text{new}}$ and $\boldsymbol{B}_{\text{new}}$ denote the matrices after removal. The objective is to compare the Frobenius norm of the error matrix, $\Delta \coloneqq \boldsymbol{P} - \boldsymbol{P}_{\text{new}} = \boldsymbol{AB} - \boldsymbol{A}_{\text{new}}\boldsymbol{B}_{\text{new}}$, under two different sparsification strategies.

A.2 Sparsification Strategies and Error Analysis

We compare two methods for selecting which column/row pair to remove.

1. Coupled Sparsification (Proposed)

Definition 1 (Coupled Sparsification)**.** *Select the index j^* that minimizes the Frobenius norm contribution of the corresponding outer product term:*

$$j^* = \underset{j \in \{1,\ldots,n\}}{\operatorname{argmin}} \|\boldsymbol{a}_j \boldsymbol{b}_j^T\|_F = \underset{j \in \{1,\ldots,n\}}{\operatorname{argmin}} \|\boldsymbol{a}_j\|_2 \|\boldsymbol{b}_j\|_2 \tag{13}$$

Remove the column $\boldsymbol{a}_{j^*}$ from $\boldsymbol{A}$ to obtain $\boldsymbol{A}_{new}^{(C)}$ and the row $\boldsymbol{b}_{j^*}^T$ from $\boldsymbol{B}$ to obtain $\boldsymbol{B}_{new}^{(C)}$. The resulting matrices have dimensions $m \times (n-1)$ and $(n-1) \times p$, respectively. The new product is implicitly defined by keeping the original pairings for the remaining terms:

$$\boldsymbol{P}_{new}^{(C)} \coloneqq \sum_{i \neq j^*} \boldsymbol{a}_i \boldsymbol{b}_i^T \tag{14}$$

Proposition 1 (Error under Coupled Sparsification)**.** *The error matrix introduced by Coupled Sparsification is exactly the removed outer product term:*

$$\Delta_1 \coloneqq \boldsymbol{P} - \boldsymbol{P}_{new}^{(C)} = \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{b}_i^T - \sum_{i \neq j^*} \boldsymbol{a}_i \boldsymbol{b}_i^T = \boldsymbol{a}_{j^*} \boldsymbol{b}_{j^*}^T \tag{15}$$

The Frobenius norm of this error is:

$$\|\Delta_1\|_F = \|\boldsymbol{a}_{j^*} \boldsymbol{b}_{j^*}^T\|_F = \|\boldsymbol{a}_{j^*}\|_2 \|\boldsymbol{b}_{j^*}\|_2 = \min_{j \in \{1,\ldots,n\}} \|\boldsymbol{a}_j\|_2 \|\boldsymbol{b}_j\|_2 \tag{16}$$

Proof. Equation (15) follows directly from the definition of $\boldsymbol{P}$ in (12) and $\boldsymbol{P}_{\text{new}}^{(C)}$. Equation (16) uses the property $\|\boldsymbol{u}\boldsymbol{v}^T\|_F = \|\boldsymbol{u}\|_2 \|\boldsymbol{v}\|_2$ and the selection criterion from (13). □

Remark 1. *Coupled Sparsification minimizes the Frobenius norm of the error matrix Δ_1 for a single pair removal, based on the greedy selection of the pair $(\boldsymbol{a}_j, \boldsymbol{b}_j)$ that contributes the least to the Frobenius norm.*

2. Individual Sparsification (Baseline)

Definition 2 (Individual Sparsification)**.** *Select the index k^* corresponding to the column of $\boldsymbol{A}$ with the smallest L2 norm, and the index l^* corresponding to the row of $\boldsymbol{B}$ (or column of $\boldsymbol{B}^T$) with the smallest L2 norm:*

$$k^* = \underset{k \in \{1,\ldots,n\}}{\operatorname{argmin}} \|\boldsymbol{a}_k\|_2 \tag{17}$$

$$l^* = \underset{l \in \{1,\ldots,n\}}{\operatorname{argmin}} \|\boldsymbol{b}_l\|_2 \tag{18}$$

Remove column $\boldsymbol{a}_{k^*}$ from $\boldsymbol{A}$ to obtain $\boldsymbol{A}_{new}^{(I)}$ and row $\boldsymbol{b}_{l^*}^T$ from $\boldsymbol{B}$ to obtain $\boldsymbol{B}_{new}^{(I)}$. The resulting matrices have dimensions $m \times (n-1)$ and $(n-1) \times p$, respectively. The new product $\boldsymbol{P}_{new}^{(I)}$ is formed by multiplying these modified matrices:

$$\boldsymbol{P}_{new}^{(I)} := \boldsymbol{A}_{new}^{(I)} \boldsymbol{B}_{new}^{(I)} \tag{19}$$

Let the columns of $\boldsymbol{A}_{new}^{(I)}$ be $\boldsymbol{a}'_p$ and the rows of $\boldsymbol{B}_{new}^{(I)}$ be $(\boldsymbol{b}'_p)^T$ for $p = 1, \ldots, n-1$. These are the original columns/rows excluding $\boldsymbol{a}_{k^*}$ and $\boldsymbol{b}_{l^*}$, implicitly re-indexed. The product is:

$$\boldsymbol{P}_{new}^{(I)} = \sum_{p=1}^{n-1} \boldsymbol{a}'_p (\boldsymbol{b}'_p)^T \tag{20}$$

Proposition 2 (Error under Individual Sparsification)**.** *The error matrix introduced by Individual Sparsification is:*

$$\Delta_2 := \boldsymbol{P} - \boldsymbol{P}_{new}^{(I)} = \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{b}_i^T - \sum_{p=1}^{n-1} \boldsymbol{a}'_p (\boldsymbol{b}'_p)^T \tag{21}$$

This error term Δ_2 can be complex, especially when $k^* \neq l^*$.

Remark 2 (Mismatch Effect)**.** *If $k^* = l^*$, then Individual Sparsification happens to select the same index. In this case, $\boldsymbol{A}_{new}^{(I)} = \boldsymbol{A}_{new}^{(C)}$ and $\boldsymbol{B}_{new}^{(I)} = \boldsymbol{B}_{new}^{(C)}$, leading to $\Delta_2 = \boldsymbol{a}_{k^*} \boldsymbol{b}_{k^*}^T$. The error norm is $\|\Delta_2\|_F = \|\boldsymbol{a}_{k^*}\|_2 \|\boldsymbol{b}_{k^*}\|_2$.*

However, if $k^* \neq l^*$, the matrix product $\boldsymbol{P}_{new}^{(I)} = \boldsymbol{A}_{new}^{(I)} \boldsymbol{B}_{new}^{(I)}$ involves multiplying columns and rows that were not originally paired. For example, if $k^* = 1, l^* = 2, n = 3$, then $\boldsymbol{A}_{new}^{(I)} = [\boldsymbol{a}_2, \boldsymbol{a}_3]$ and $\boldsymbol{B}_{new}^{(I)} = \begin{bmatrix} \boldsymbol{b}_1^T \\ \boldsymbol{b}_3^T \end{bmatrix}$. The product is $\boldsymbol{P}_{new}^{(I)} = \boldsymbol{a}_2 \boldsymbol{b}_1^T + \boldsymbol{a}_3 \boldsymbol{b}_3^T$. The original product was $\boldsymbol{P} = \boldsymbol{a}_1 \boldsymbol{b}_1^T + \boldsymbol{a}_2 \boldsymbol{b}_2^T + \boldsymbol{a}_3 \boldsymbol{b}_3^T$. The error is $\Delta_2 = \boldsymbol{P} - \boldsymbol{P}_{new}^{(I)} = \boldsymbol{a}_1 \boldsymbol{b}_1^T + \boldsymbol{a}_2 \boldsymbol{b}_2^T - \boldsymbol{a}_2 \boldsymbol{b}_1^T$.

In general, Δ_2 can be expressed as:

$$\Delta_2 = \underbrace{\boldsymbol{a}_{k^*} \boldsymbol{b}_{k^*}^T}_{\textit{Term related to removed } \boldsymbol{a}_{k^*}} + \underbrace{\boldsymbol{a}_{l^*} \boldsymbol{b}_{l^*}^T}_{\textit{Term related to removed } \boldsymbol{b}_{l^*}} + \underbrace{\left(\sum_{i \neq k^*, l^*} \boldsymbol{a}_i \boldsymbol{b}_i^T - \boldsymbol{P}_{new}^{(I)} \right)}_{\textit{Mismatch Effect (if } k^* \neq l^*)} \tag{22}$$

The "Mismatch effect" arises because the structure $\sum \boldsymbol{a}_i \boldsymbol{b}_i^T$ is broken by removing components based on potentially different indices k^* and l^*. Calculating $\boldsymbol{P}_{new}^{(I)}$ involves multiplying the remaining $n-1$ columns of $\boldsymbol{A}_{new}^{(I)}$ with the remaining $n-1$ rows of $\boldsymbol{B}_{new}^{(I)}$, creating cross-terms that differ from the original summation structure. This structure makes $\|\Delta_2\|_F$ difficult to analyze directly and prevents it from directly optimizing a simple objective related to $\|\boldsymbol{a}_{k^*}\|_2$ or $\|\boldsymbol{b}_{l^*}\|_2$.

A.3 Theoretical Comparison

Proposition 3 (Probabilistic Error Comparison)**.** *The error introduced by Coupled Sparsification, $\|\Delta_1\|_F$, is probabilistically likely to be smaller than the error introduced by Individual Sparsification, $\|\Delta_2\|_F$.*

Argument Sketch. 1. Coupled Sparsification, by definition (13), selects the pair (j^*) such that the Frobenius norm of the removed term, $\|\boldsymbol{a}_{j^*} \boldsymbol{b}_{j^*}^T\|_F$, is minimized. This minimized value is precisely the Frobenius norm of the error, $\|\Delta_1\|_F$ (Equation (16)). The method directly optimizes an upper bound related to the reconstruction error for the specific structure $\boldsymbol{A}\boldsymbol{B} = \sum \boldsymbol{a}_i \boldsymbol{b}_i^T$.

2. Individual Sparsification selects indices k^* and l^* based on minimizing individual vector norms $\|\boldsymbol{a}_k\|_2$ and $\|\boldsymbol{b}_l\|_2$ independently. - If $k^* = l^*$, the error norm is $\|\Delta_2\|_F = \|\boldsymbol{a}_{k^*}\|_2 \|\boldsymbol{b}_{k^*}\|_2$. Since j^* minimizes the product $\|\boldsymbol{a}_j\|_2 \|\boldsymbol{b}_j\|_2$, we have $\|\Delta_1\|_F \leq \|\Delta_2\|_F$ in this specific case. - If $k^* \neq l^*$, the error Δ_2 includes the complex "Mismatch effect" described in Remark 2. The selection criteria (17) and (18) do not directly minimize $\|\Delta_2\|_F$. The mismatch term introduces components unrelated to the individual norms being minimized, breaking the direct link between the optimization objective and the resulting error norm.

3. Because Coupled Sparsification directly minimizes the quantity $\|\boldsymbol{a}_j\|_2\|\boldsymbol{b}_j\|_2$ which equals the error norm $\|\Delta_1\|_F$, while Individual Sparsification minimizes separate quantities ($\|\boldsymbol{a}_k\|_2$, $\|\boldsymbol{b}_l\|_2$) leading to a complex error Δ_2 (often involving mismatch effects) that isn't directly minimized, it is probabilistically expected that $\|\Delta_1\|_F \leq \|\Delta_2\|_F$. The individual strategy optimizes components in isolation, failing to account for their coupled contribution to the product $\boldsymbol{AB}$ and the structure of the resulting error upon removal, especially when $k^* \neq l^*$. □

A.4 Empirical Validation

To verify this theoretical advantage, empirical tests were conducted.

1. Random matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ were generated for various dimensions ($n \times n$ with $n \in \{100, 500, 1000, 5000\}$).
2. For each dimension, 5000 random matrix pairs were generated.
3. The Frobenius norm errors, $\|\Delta_1\|_F$ (Coupled) and $\|\Delta_2\|_F$ (Individual), were computed after removing one column/row pair using each strategy.
4. The probability $\mathbb{P}(\|\Delta_1\|_F < \|\Delta_2\|_F)$ was estimated from the trials.

The results are summarized in Table 7.

Table 7: Empirical Comparison of Frobenius Error Norms

Matrix Size (n)	Avg. $\|\Delta_1\|_F$ (Coupled)	Avg. $\|\Delta_2\|_F$ (Individual)	$\mathbb{P}(\|\Delta_1\|_F < \|\Delta_2\|_F)$ (%)
100×100	76.4	757.9	99.0
500×500	434.4	8498.9	99.7
1000×1000	900.5	23931.2	99.9
5000×5000	4743.7	266465.9	100.0

The empirical results strongly support the theoretical analysis, showing that Coupled Sparsification yields significantly lower error with very high probability ($> 99\%$) compared to Individual Sparsification across different matrix dimensions.

B Theoretical Motivation for Coupled Sparsification in FFNs

B.1 Derivation from Pruning Error Approximation (for GELU)

Our method, **Coupled Sparsification**, removes the j-th column of $\mathbf{W}_{in}$ and the j-th row of $\mathbf{W}_{out}$ together. The true error this introduces for an input $\mathbf{X}$ is:

$$\text{Error}(j, \mathbf{X}) = \|\text{GELU}(\mathbf{X}\mathbf{W}_{in})\mathbf{w}_{out,j}\|_F \tag{23}$$

To analyze this without input-dependency, we approximate the error using the first-order Taylor expansion of GELU around zero (GELU$(u) \approx c \cdot u$). Applying this approximation yields a bound on the true error:

$$\begin{aligned}
\text{Error}(j, \mathbf{X}) &\approx \|c \cdot (\mathbf{X}\mathbf{w}_{in,j})\mathbf{w}_{out,j}\|_F &(24)\\
&= c \cdot \|\mathbf{X}(\mathbf{w}_{in,j}\mathbf{w}_{out,j})\|_F &(25)\\
&\leq c \cdot \|\mathbf{X}\|_F \cdot \|\mathbf{w}_{in,j}\mathbf{w}_{out,j}\|_F &(26)\\
&= (c \cdot \|\mathbf{X}\|_F) \cdot (\|\mathbf{w}_{in,j}\|_2 \cdot \|\mathbf{w}_{out,j}\|_2) &(27)
\end{aligned}$$

This derivation shows that the true error is bounded by a term proportional to our **Coupled Metric**. The data-dependent part is a common scaling factor, leaving our metric as the decisive factor for ranking. This justifies why the **Coupling Effect** exists: the coupled weight matrix drives the dominant, first-order term of the FFN computation.

B.2 Applicability to Other FFN Types and the Coupled Effect

While the mathematical forms of other FFN layer types, such as ReLU or SwiGLU, differ from GELU and may not present a direct linear term in the same manner, our approach is designed to adapt to these structures. A key insight is that for many non-linear activations (e.g., ReLU, SiLU, GELU), significant input magnitudes can lead to negligible outputs (e.g., for negative inputs).

Our method addresses this by focusing on the *coupled influence* of the input and output weight matrices. For SwiGLU, as elaborated in Footnote 2, we establish a coupled structure by defining $\mathbf{W}_1 = \mathbf{W}_{\text{gate}} \odot \mathbf{W}_{\text{up}}$ and $\mathbf{W}_2 = \mathbf{W}_{\text{down}}$. This formulation is chosen to capture the critical functional relationships and information flow between these interdependent weight matrices. By assessing importance through this **coupled effect**, we inherently account for how the non-linearity modulates a neuron's ultimate contribution, leading to a more accurate importance evaluation.

B.3 Coupled Sparsification V.S. Uncoupled Sparsification in FFN

To empirically validate the effectiveness of our **Coupled Sparsification** method for FFN matrices, we conducted direct comparisons against **Uncoupled Sparsification** on both LLaMA2 (with SwiGLU) and GPT-2 (with GELU) models, evaluated on the WikiText-2 dataset. As presented in Table 8, Coupled Sparsification dramatically outperforms Uncoupled Sparsification.

Table 8: Coupled Sparsification V.S. Uncoupled Sparsification in FFN.

	Sparsity	30%	50%	70%	90%
LLaMA2 (SwiGLU)	Coupled	142.2	730.7	2115.6	6481.6
(PPL=9.4)	Uncoupled	187062.7	208483.4	261173.1	173291.2
GPT2-m (GELU)	Coupled	19137.1	24296.2	16157.3	28261.5
(PPL=21.4)	Uncoupled	4205783.8	1158156.8	12031555.3	7131925831.1

C Hyper-parameters

C.1 Selection of K and θ.

As described in Algorithm 1, K determines the percentage of $\boldsymbol{W}_{couple}$'s that will be sparsified in each epoch, and threshold θ impacts the amount of to-be-removed coupled row/column pairs within those unimportant $\boldsymbol{W}_{couple}$'s. As shown in Fig. 3, when applying EcoSpa at the pre-training stage, larger θ brings better training performance, while the model is less sensitive to the change of K. On the other hand, Fig. 4 shows that it is better to use smaller K at the fine-tuning stage, while the selection of θ is less significant in this scenario. We hypothesize that such difference might be due to the existence of a pre-trained model in the fine-tuning process since a pre-trained model typically has more diverse distribution of $\boldsymbol{W}_{couple}$ than the model being sparsely trained from random initialization, making the identification of unimportant $\boldsymbol{W}_{couple}$ more effective. Therefore, considering K cannot be too small (otherwise, it is challenging to meet the model size budget (see the change of parameters (dashed curve) in Fig. 4), we set $K = 30\%$ and $\theta = 90\%$ in our experiments.

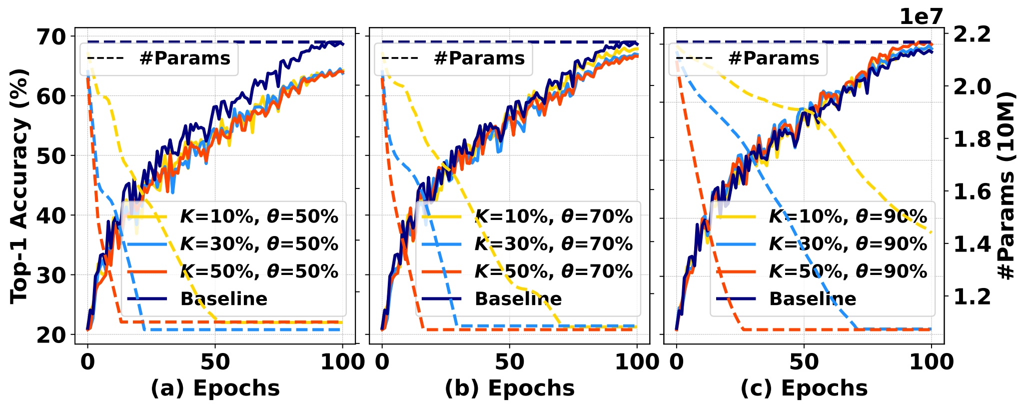

Figure 3: Pre-training sparse DeiT-Small on CIFAR-10 with various K and θ. The target model size is 11M.

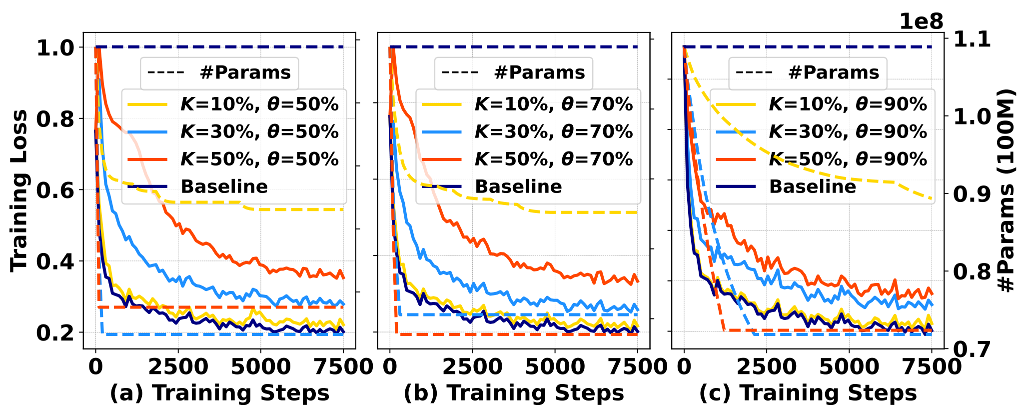

Figure 4: Fine-tuning Pre-trained sparse BERT-Base on SQuAD with various K and θ. The target model size is 72.6M.

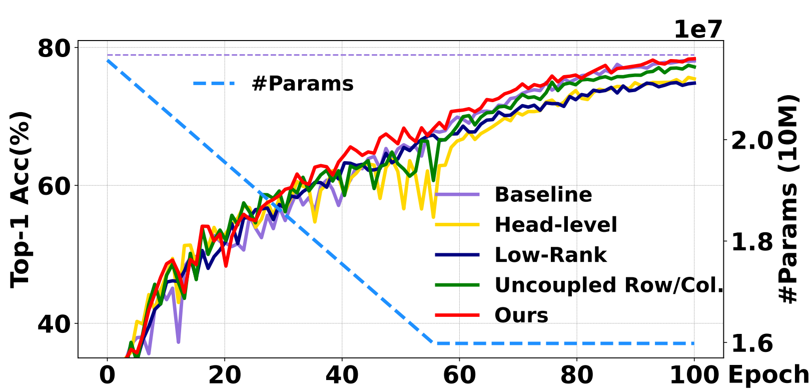

Figure 5: Pre-training DeiT-Small model on CIFAR-10 dataset using different $\boldsymbol{W}_{couple}$ compression methods. All the methods remove the same number of parameters within the same $\boldsymbol{W}_{couple}$'s in each epoch.

C.2 Importance of Coupled Sparsification.

Fig. 5 compares the training performance using different methods to compress $\boldsymbol{W}_{couple}$. It is seen that our proposed coupled row/column-wise sparsification scheme achieves the best performance. In particular, it outperforms the uncoupled row/column-based solution, demonstrating the importance of removing the row/column of $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ in a coupled way. Notice that though our approach is inspired by tSVD, it achieves better performance than directly applying tSVD on $\boldsymbol{W}_{couple}$. This is because SVD not only changes the structure of the model but also alters the numerical distribution of the original model weights. Consequently, the optimizers that keep track of moment information, *e.g.*, Adam [33], cannot work well since they will not be able to use previously accumulated information, thereby affecting the model performance.

Table 9: Overview of Experimental Setups and Hyper-parameters

Experiments	**Model**	**Dataset**	**top -** $K(\%)$	$\theta(\%)$
Table 1	LLaMA-1B	C4	30	90
Table 2	LLaMA-7B	C4	30	90
Table 3	GPT	WikiText	30	90
Table 4	DeiT	ImageNet	30	90
Table 5	LLaMA2-7B@80%	WikiText	30	95
Table 5	LLaMA2-7B@70%	WikiText	30	90
Table 5	LLaMA2-7B@60%	WikiText	30	90
Table 5	LLaMA2-7B@50%	WikiText	30	85
Table 6	LLaMA2-7B	WikiText	30	95

C.3 Observation of Compressed Model.

We analyze the resulting structure of the compressed models. Intriguing patterns emerge, revealing how the coupled sparsification strategy interacts differently with components based on model architecture and task. These observations, detailed below and summarized in Tables 10 and 11, provide insights into the method's adaptive nature.

- **Language Models (e.g., GPT-2):** In language models like GPT-2, the primary goal is sequence modeling and maintaining contextual coherence. We observe that EcoSpa tends to preserve the dimensions of the value projection ($\boldsymbol{W}^V$) and the final output projection ($\boldsymbol{W}^O$) matrices within the attention blocks. These components are crucial for integrating contextual information and propagating it through the network. Conversely, the query ($\boldsymbol{W}^Q$) and key ($\boldsymbol{W}^K$) matrices, which predominantly determine the attention patterns (implicitly via $\boldsymbol{W}^Q \boldsymbol{W}^{K^T}$), undergo more aggressive compression. This suggests that EcoSpa identifies greater redundancy within the attention pattern formation mechanism while prioritizing the preservation of the value pathway for maintaining coherence, aligning with findings on functional roles within transformer circuits [18].

- **Vision Models (e.g., DeiT):** In vision transformers aimed at classification tasks like DeiT, a different pattern emerges. While EcoSpa compresses matrices throughout most of the network layers (Blocks 1-10 in DeiT-Tiny, see Table 10), the dimensions in the final, higher-level blocks (Blocks 11-12) often remain largely unchanged or are compressed less aggressively. These later layers are typically responsible for consolidating abstract features critical for the final classification decision. The observed behavior indicates that EcoSpa implicitly safeguards these high-level representations by reducing compression in deeper layers, while readily exploiting redundancies in the earlier feature extraction layers.

These distinct behaviors across model types underscore EcoSpa's ability to adaptively apply sparsity based on the implicit functional importance of different components, preserving critical pathways while effectively pruning less essential dimensions identified through the coupled analysis.

Table 10: Dimensionality changes in DeiT-Tiny (3 Heads per block) after applying EcoSpa block-wise.

Layer / Block	$\boldsymbol{W}^Q, \boldsymbol{W}^K$ Dim.	$\boldsymbol{W}^V, \boldsymbol{W}^O$ Dim.	$\boldsymbol{W}^{\text{in}}, \boldsymbol{W}^{\text{out}}$ (FFN) Dim.
Original	192×64	192×64	192×768
After EcoSpa			
Block 1	192×37	192×41	192×399
Block 2	192×39	192×43	192×393
Block 3	192×40	192×42	192×501
Block 4	192×41	192×50	192×451
Block 5	192×43	192×50	192×419
Block 6	192×49	192×50	192×402
Block 7	192×56	192×50	192×391
Block 8	192×55	192×56	192×379
Block 9	192×56	192×57	192×366
Block 10	192×50	192×56	192×387
Block 11	192×64	192×64	192×768
Block 12	192×64	192×64	192×768

Table 11: Dimensionality changes in GPT-2-Small (12 Heads per block) after applying EcoSpa block-wise.

Layer / Block	$\boldsymbol{W}^Q, \boldsymbol{W}^K$ Dim.	$\boldsymbol{W}^V, \boldsymbol{W}^O$ Dim.	$\boldsymbol{W}^{\text{in}}, \boldsymbol{W}^{\text{out}}$ (FFN) Dim.
Original	768×64	768×64	768×3072
After EcoSpa			
Block 1	768×64	768×64	768×2064
Block 2	768×25	768×64	768×1337
Block 3	768×21	768×64	768×1259
Block 4	768×21	768×64	768×1643
Block 5	768×37	768×64	768×1639
Block 6	768×30	768×64	768×1613
Block 7	768×20	768×64	768×1629
Block 8	768×20	768×64	768×1104
Block 9	768×20	768×64	768×1073
Block 10	768×21	768×64	768×1589
Block 11	768×26	768×64	768×1574
Block 12	768×21	768×64	768×2044

D Broader Impacts

Our research on sparse training for transformer models enables more computationally efficient and environmentally sustainable training and deployment of AI models. Developing sparse training methods for Transformer models can significantly accelerate AI training and inference, fostering innovation while reducing energy consumption. This not only democratizes the development and utilization of AI but also aligns with global efforts towards sustainable computing practices, mitigating the environmental impact associated with training resource-intensive neural networks. As AI permeates various domains, optimizations like sparse training will play a crucial role in striking a balance between model performance and environmental responsibility, ensuring the responsible advancement of this technology.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Refer to Abstract section.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We demonstrated that the proposed sparse transformer training approach brings significant efficiency and performance improvement. However, due to the limited computing resources, we do not conduct experiments on larger models (*e.g.*, 65B and 175B), especially in the pre-training scenario. A more comprehensive exploration will be the subject of our future work.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Appendix

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Appendix

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide training settings for all experiments and we plan to release our code after the conference.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Appendix

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Refer to the Experiments section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In the experimental section, we have indicated the model of the workstation used for training the model, as well as the training time.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We strictly adhere NeurIPS Code of Ethics.646

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: This paper focuses on how to boost the speed of model training and inference, which has significant benefits both in terms of environmental protection and the cost for model deployment.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification:

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.