ResMoE: Parameter Compression for Pre-Trained Mixture-of-Experts Transformer via Residual Restoration

Anonymous Authors[1]

Abstract

Mixture-of-Experts (MoE) Transformer, the backbone architecture of multiple recent phenomenal language models, leverages sparsity by activating only a fraction of model parameters for each input. The sparse structure, while allowing constant time cost, results in space inefficiency: model inference is now hindered by the storage and memory bandwidth demands. Addressing this issue with current hardware architecture is challenging and under-explored. We introduce ResMoE, an innovative MoE compression framework that utilizes Wasserstein barycenter to extract a common expert (barycenter expert) and applies unstructured pruning to approximate the residuals between this barycenter expert and the other ones. This approach is optimized for modern hardware, avoiding the complexities of sparse matrix multiplication. ResMoE enhances the space efficiency for inference of pre-trained MoE Transformers in a task-agnostic, one-shot manner without retraining while minimizing the accuracy loss, thereby facilitating broader language model accessibility. We demonstrate the effectiveness of ResMoE through extensive experiments on Switch Transformer, Mixtral, and DeepSeekMoE models, by pruning up to 75% of parameters of the specific layers with negligible performance loss. The code and model weights will be open-sourced upon publication.

1. Introduction

The transformative impact of the Transformer architecture in the domain of machine learning is undeniable, for both the field of natural language processing (Vaswani et al., 2017; Devlin et al., 2019; Fedus et al., 2022; Anil et al., 2023; OpenAI, 2022; Raffel et al., 2023) and computer vision (Liu et al., 2021; Wang et al., 2021; Fan et al., 2021). To further improve the capabilities of Transformer-based pre-trained large language models (LLMs), one general strategy is to scale up their parameters. Mixture-of-Experts (MoE) (Shazeer et al., 2017) extends the traditional feedforward neural network (FFN) layer by replacing a single multilayer perceptron (MLP) with multiple MLPs, referred to as 'experts'. While enhancing the performance, sparse MoE keeps computing costs (FLOPs) comparable to the original dense model, as only a few selected experts will be activated each time. The framework of an MoE layer is demonstrated in Fig. 1. Switch Transformer (Fedus et al., 2022) exemplifies this approach by expanding the T5 model (Raffel et al., 2023) to an MoE structure, scaling it up to at most 2048 times the size of the original dense T5 model. Similarly, Mixtral (Jiang et al., 2024) upscales Mistral 7B (Jiang et al., 2023) to an 8×7B MoE structure, achieving performance that matches or even surpasses that of Llama2 70B (Touvron et al., 2023). DeepSeekMoE (Dai et al., 2024) utilizes fine-grained experts compared to the other structures, featuring 64 experts per layer.

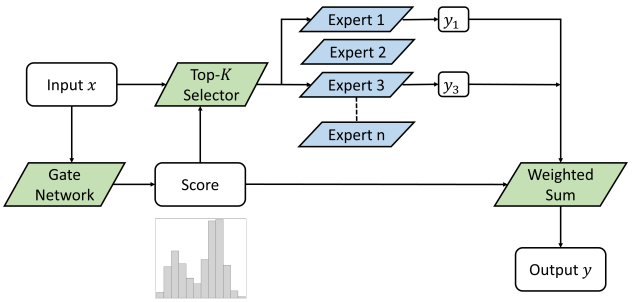

Figure 1. An illustrative example of an MoE layer. In this example, the Top-K Selector, along with the Gate Network—often referred to as the 'router'—selects Experts 1 and 3 based on their scores for the given input.

However, the enormous number of the parameters has now become the bottleneck for MoE Transformers, since they require much more GPU memory to load the model. Even in the smallest base Switch Transformer, the parameter count of each expert is 4.7M, and the expert size for Mixtral

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

even reaches 176.2M. Furthermore, the presence of 8 and even more experts in each layer exacerbates the memory demands, bringing a strong need to compress the experts in the MoE structure.

To leverage the capabilities of MoE LLMs, several research avenues have been pursued. For example, one approach is model fusion (Singh & Jaggi, 2020; Ainsworth et al., 2023), which involves combining multiple general MLPs that can be adapted to experts as well. However, this fusion operation is executed on a layer-by-layer basis, which will incur extra runtime for permutation operations. More recently, various studies have introduced the concept of expert merging (He et al., 2023b; Li et al., 2023a), as a method to reduce the number of experts within each layer of the MoE model. Nevertheless, this global reduction in the number of experts potentially may lead to a substantial loss of the specialized knowledge that individual experts possess (a toy analysis is provided in Section 4.1).

To address the above problems, we introduce a novel MoE compression framework named ResMoE. Our approach capitalizes on exploiting the inherent parameter redundancy in MoE models by employing the Wasserstein barycenter technique, which formulates a distributional representation of experts to extract their common characteristics. Subsequently, we employ unstructured pruning to approximate the residual matrices between this barycenter expert and each specific expert in a hardware-friendly fashion. In summary, the contribution of this work is three-fold:

- We introduce Wasserstein barycenters and residual restoration into MoE compression, aiming to maintain the distinctive attributes of each expert with fewer parameters.
- We propose ResMoE, a novel MoE Transformer compression framework that aims to improve memory efficiency in a one-shot and task-agnostic manner, with no extra training required.
- We validate ResMoE through extensive experiments on the encoder-decoder Switch Transformer model, as well as on the decoder-only models, Mixtral and DeepSeek-MoE. Our results demonstrate that ResMoE can prune up to 75% of the parameters with only marginal performance loss, verifying the effectiveness and the versatility of it.

2. Related Work

Mixture-of-Expert (MoE) Transformer Compression. Rather than applying existing compression techniques individually to the expert MLP, MC-SMoE (Li et al., 2023a), MEO (He et al., 2023b) and OneS (Xue et al., 2022) merge the experts into smaller groups, reducing the count of the experts. However, due to the high complexity of deciding which experts to retain, reducing the experts could lead to a substantial loss of information. Gao et al. (2022) instead turn to keep each expert, divide them into several sections, and share the core section among them. This method provides a limited contribution to the MoE compression technique since their goal is to train a new MoE like structure from scratch, instead of compressing an existing one.

Alternatively, fusion-based methods (Singh & Jaggi, 2020; Anil et al., 2023) employ the principles of permutation and optimal transport. Originally proposed for consolidating distinct models into a single entity, these methods can be dynamically adapted for consolidating MoE's experts. Despite utilizing optimal transportation and permutation transport matrices, they are implemented layer-wisely, which illogically apply the permutations derived from adjacent layers to the current one. Moreover, these methods incur overhead due to the extra time required for permutation operations.

General Model Compression Techniques. The predominant focus of machine learning model compression research primarily involves system-level optimization. Quantization aims at hardware efficiency by reducing model weight bit-depth from 32-bit floating point (FP32) to 8-bit integers (INT8) (Bhandare et al., 2019; Dettmers et al., 2022) or even lower bits (Li et al., 2022; Tao et al., 2022; Frantar et al., 2023). Our focus, however, is on reducing the parameter count of the MoE model, making quantization methods not directly related to our approach.

Additionally, knowledge distillation (Gou et al., 2021) seeks to transfer knowledge from LLMs to smaller models, while this approach necessitates extensive retraining involving both the original LLM and the compact model. Truncated singular value decomposition (SVD) (Denton et al., 2014b) has been used to streamline CNNs by reducing redundancy through linear structure exploitation within networks, yet it faces limits in representational capacity, often leading to decreased performance due to overly huge dimension reduction. Pruning techniques (Liu et al., 2018), evolving with the Lottery Tickets Hypothesis (Frankle & Carbin, 2018, LTH), seek efficient sub-networks within larger models but require extensive retraining to maintain accuracy. While some one-shot pruning methods (Wang et al., 2020) do exist, they remain computationally expensive and do not consider the structure of MoE, raising concerns about whether such methods can ensure that the compressed models retain their effectiveness for downstream tasks.

3. Preliminaries and Notation

This section provides the background information on MoE and optimal transport, along with Wasserstein barycenter.

3.1. Mixture-of-Experts Modules

Throughout this paper, we consider the classical setting for the ease of analysis, where each expert takes the form of

a multilayer perceptron (MLP) in a feed-forward network (FFN) sub-layer of a Transformer. It is worth noting that our method can be readily extended to different types of expert network architectures. Each Mixture-of-Expert (MoE) layer comprises of N experts. The k-th expert E_k (a function to transform input vector $\mathbf{x}$ to a new feature) in an FFN sub-layer is denoted as:

$$E_k(\mathbf{x}) = \mathbf{W}_k^{(2)}\sigma\left(\mathbf{W}_k^{(1)}\mathbf{x} + \mathbf{b}_k^{(1)}\right) + \mathbf{b}_k^{(2)},$$

where we apply element-wise activation function $\sigma(\cdot)$ to the input $\mathbf{x} \in \mathbb{R}^p$, and $(\mathbf{W}_k^{(1)}, \mathbf{b}_k^{(1)}) \in \mathbb{R}^{p_{\text{I}} \times (p+1)}$, $(\mathbf{W}_k^{(2)}, \mathbf{b}_k^{(2)}) \in \mathbb{R}^{p \times (p_{\text{I}}+1)}$ are respectively the weight matrices and bias vectors in the linear transforms of the MLP (with input/output dimension p and inner dimension p_{I}). The output of the MoE layer is given by: $\sum_{k=1}^N [G(\mathbf{x})]_k \cdot E_k(\mathbf{x})$. Here $G(\mathbf{x}) = \text{Softmax}\left(\text{TopK}\left(\mathbf{W}_g\mathbf{x}\right)\right)$ returns the normalized sparse router gating vector for all experts, where $\text{TopK}(\mathbf{g}_i) = \mathbf{g}_i$ when $\mathbf{g}_i$ is within the top-k values of $\mathbf{g} \in \mathbb{R}^N$, otherwise $\text{TopK}(\mathbf{g}_i) = -\infty$; $\mathbf{W}_g \in \mathbb{R}^{N \times p}$ represents the linear transform, turning the input x into the logit for each expert. The whole framework of the MoE layer is shown in Fig. 1. The space bottleneck comes from the large number of experts (ranging from 8 to 64 and even more (Fedus et al., 2022; Dai et al., 2024)) and the tremendous size of the weight matrices in each expert (e.g., 176.2M parameters for each expert in Mixtral (Jiang et al., 2024)). The sparse design renders the total number of parameters redundant compared to the base dense model. In this paper, we aim to address the redundancy problem while retaining effectiveness.

3.2. Optimal Transport, Wasserstein Barycenter, and Their Applications to Model Fusion

Optimal transport (OT) theory has achieved great success in depicting the underlying geometry of distributions (Peyre & Cuturi, 2020). We consider two distributions $\mu = \sum_{i=1}^n \boldsymbol{\alpha}_i \delta_{x_i}$ and $\nu = \sum_{j=1}^m \boldsymbol{\beta}_j \delta_{y_j}$ with $\boldsymbol{\alpha}_i, \boldsymbol{\beta}_j$ as masses respectively assigned to points x_i, y_j, and δ_x being the Dirac unit mass located on x. (Along this paper, μ, ν will always be the distributions of discrete random variables.) OT reflects a process of transporting the mass from positions x_i's to y_j's (transforming the source distribution μ to the target distribution ν) with minimal overall cost, in which the cost of transporting a unit mass from x_i to y_j is given by the cost function $D(x_i, y_j)$.

A transport plan can be specified by a matrix $\mathbf{M} \in \mathbb{R}^{n \times m}$, where $\mathbf{M}_{i,j}$ indicates the mass to be transported from x_i to y_j. We note that the column and row sums of $\mathbf{M}$ respectively equal to $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, implying all the masses in μ are transported to the desired points in ν, i.e., $\mathbf{M} \in \Pi(\boldsymbol{\alpha}, \boldsymbol{\beta}) := \{\mathbf{M} \in \mathbb{R}^{n \times m} | \sum_{j=1}^m \mathbf{M}_{i,j} = \boldsymbol{\alpha}_i, \sum_{i=1}^n \mathbf{M}_{i,j} = \boldsymbol{\beta}_j\}$. OT seeks the optimal plan to transport μ to ν w.r.t the overall

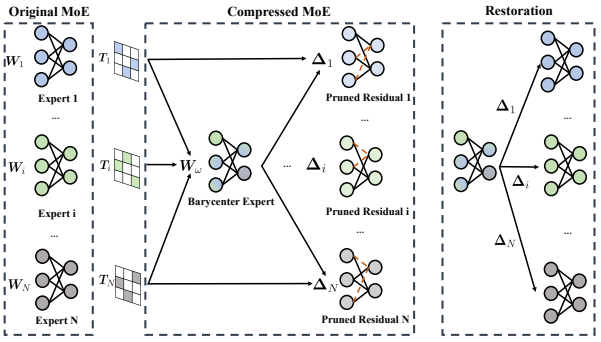

Figure 2. The overall framework of ResMoE. The orange dashed lines denote the connections within the network are pruned. We introduce permutation matrices $\mathbf{T}$ to obtain the barycenter expert $\mathbf{W}_\omega$ from a distribution view. Instead of compressing the original experts directly, we opt to prune the residual matrices ($\boldsymbol{\Delta}$, illustrated with lighter colors) between each expert and the barycenter expert, with the capability to dynamically and efficiently restore the original matrices during the inference phase.

transportation cost, formulated as (Peyre & Cuturi, 2020)

$$\text{OT}(\mu, \nu) := \underset{\mathbf{M} \in \Pi(\boldsymbol{\alpha}, \boldsymbol{\beta})}{\arg\min} \sum_{i,j} \mathbf{M}_{i,j}\mathbf{C}_{i,j} = \underset{\mathbf{M} \in \Pi(\boldsymbol{\alpha}, \boldsymbol{\beta})}{\arg\min} \langle \mathbf{M}, \mathbf{C} \rangle,$$

where $\mathbf{C} := [D(x_i, y_j)]_{ij} \in \mathbb{R}^{n \times m}$ is the cost matrix.

When we set the cost function as $D(x_i, y_j) = \|x_i - y_j\|^2$, we can obtain 2-Wasserstein distance as

$$W_2^2(\mu, \nu) := \min_{\mathbf{M} \in \Pi(\boldsymbol{\alpha}, \boldsymbol{\beta})} \langle \mathbf{M}, \mathbf{C} \rangle .$$

In this paper, we specifically focus on the free-support Wasserstein barycenter problem induced by 2-Wasserstein distance. Given a set of distributions $\mu_1, \ldots, \mu_N$, the Wasserstein barycenter $\bar{\mu}$ is the "average" distribution in terms of the Wasserstein distance. To regulate the form of $\bar{\mu}$ in numerical computation, we specify $\bar{\mu}$ as a uniform distribution on n points, and optimize it through:

$$\bar{\mu} = \underset{\{x_i\}_{i=1}^n}{\arg\min} \frac{1}{N} \sum_{k=1}^N W_2^2\left(\mu_k, \sum_{i=1}^n \frac{1}{n}\delta_{x_i}\right). \quad (1)$$

We comment Cuturi & Doucet (2014) have provided efficient numerical algorithms for free-support Wasserstein barycenter problems, which will be heavily utilized in our implementations. In a nutshell, we conclude Wasserstein barycenter captures the underlying advection on the distribution space, offering a powerful tool for aggregating distributions in a geometrically meaningful way.

Alignment-based model fusion (Singh & Jaggi, 2020; Ainsworth et al., 2023) is proposed to fuse multiple general MLPs (which may have more than two layers) and relates to OT. Specifically in the work of OT Fusion (Singh & Jaggi, 2020), for the first layer in an two-layer MLP (for example),

their algorithm takes the weights $(\mathbf{W}_k^{(1)}, \mathbf{b}_k^{(1)})$ in each expert E_k and in E_ω as the source distributions and the target distribution, respectively. They directly take $\mathbf{W}_k^{(1)}$ as the design points (empirical distributions) and then regard their free-support Wasserstein barycenter as the common pattern extraction of the first layer in each expert, returning $\mathbf{W}_\omega^{(1)}$ and the corresponding permutation matrix $\mathbf{T}_k^{(1)}$'s (obtained from the transport matrices) for $\mathbf{W}_k^{(1)}$. Accordingly, they then pre-align the second layers as $\mathbf{W}_k^{(2)}\mathbf{T}_k^{(1)}$, repeat the procedure above and similarly obtain the extracted layer $\mathbf{W}_\omega^{(2)}$ and the permutation matrices $\mathbf{T}_k^{(2)}$. Similarly, in Git Re-basin (Ainsworth et al., 2023), they propose to align the weights through a greedy loop for each layer, which demonstrates zero-barrier linear mode connectivity (Frankle et al., 2020) between independently trained models on the same dataset.

As a closing remark, we note the alignment-based model fusion technique implies a layer-by-layer strategy to formulate MoE layers as distributions and to merge experts. We will shortly analyze the strategy in Section 4.1 and empirically evaluate its performance in Section 5.

4. Proposed Methodology

In this section, we first analyze the limitations of previous works, followed by a detailed introduction of ResMoE, with its visualization provided in Figure 2. For a comprehensive visual comparisons between ResMoE and previous baseline methods, please refer to Appendix C.1 and Figure 4.

4.1. Limitations of Previous Works

Layer-by-layer Fusion Strategy. We first illustrate how to adapt the layer-by-layer strategy in OT fusion (Singh & Jaggi, 2020) to expert restoration, following the notations in the previous section. To recover the k-th expert E_k, their barycenter expert needs to be transformed as $(\mathbf{T}_k^{(2)})^\top E_\omega(x)$, to fix the order of the output elements. We remark that the permutation $(\mathbf{T}_k^{(2)})^\top$ for the output, along with the pre-alignment $\mathbf{W}_k^{(2)}\mathbf{T}_k^{(1)}$ are impractical since they ignore the inherent discord between the layers' transformation characteristics. Layers in an MLP capture different features from each other, which means that directly aligning the layers using the pre-alignment transformation $\mathbf{W}_k^{(2)}\mathbf{T}_k^{(1)}$may not be reasonable. Each layer in an MLP has a specific role in transforming the input features, and aligning them without considering their individual contributions can lead to suboptimal results. The strategy also suffers from additional runtime for the permutation during algorithm execution, especially when dealing with multiple layers and experts. For example, in the case of Mixtral, which has three layers in one MLP, pre-alignment would require two extra operations per MLP. The additional computations introduced by pre-alignment can make the algorithm less practical and efficient, especially when scaling to larger models with many experts.

Similarly, Git Re-basin (Ainsworth et al., 2023) requires calculating and storing a permutation matrix $\mathbf{T}_k^{(l)}$ to align the current layer l with layer $l-1$ and layer $l+1$, with the pre-alignment of $\mathbf{W}_k^{(l)}\mathbf{T}_k^{(l-1)}$ and $\mathbf{W}_k^{(l+1)}\mathbf{T}_k^{(l)}$; it thus has the same limitation as OT Fusion in regards to both impractical pre-alignment and additional runtime.

Expert Merging Strategy (He et al., 2023b; Li et al., 2023a). Several studies recently introduced expert merging to reduce the number of experts in MoE layers. In general, this strategy merges experts relying on expensive information such as router gating score distribution or router activation frequency for the given task. However, the direct reduction in the number of experts may lead to a huge deviation from the original module output, especially in the zero-shot setting we consider. To better understand this, we first revisit the MoE layer from a comprehensive, all-experts matrix view. The router matrix $\mathbf{R}$ can be defined as:

$$\operatorname{diag}(G(\mathbf{x})) \otimes \mathbf{I}_{p_\mathrm{I}} = \begin{bmatrix} [G(\mathbf{x})]_1\mathbf{I} & & \\ & \cdots & \\ & & [G(\mathbf{x})]_k\mathbf{I} \\ & & \cdots \end{bmatrix}_{N\cdot p_\mathrm{I} \times N\cdot p_\mathrm{I}},$$

where $\otimes$ is the Kronecker product and $\mathbf{I}$ is the identity matrix, and $G(\mathbf{x})$ being a sparse vector. The weight matrices in the MoE layer are:

$$\mathbf{W}^{(1)} = \begin{pmatrix} \mathbf{W}_1^{(1)} & \cdots & \mathbf{W}_N^{(1)} \end{pmatrix}^\top_{N\cdot p_\mathrm{I} \times p},$$
$$\mathbf{W}^{(2)} = \begin{pmatrix} \mathbf{W}_1^{(2)} & \cdots & \mathbf{W}_N^{(2)} \end{pmatrix}_{p \times N\cdot p_\mathrm{I}}.$$

The output of the MoE layer can accordingly be expressed as (omitting the bias term for simplicity):

$$\mathbf{y} = \mathbf{W}^{(2)}\mathbf{R}\sigma\left(\mathbf{W}^{(1)}\mathbf{x}\right), \tag{2}$$

in which $\mathbf{R}$ encapsulates crucial expert knowledge and exhibits high sparsity, as typically only a selected number of experts are activated within each MoE layer.

To analyze the difficulty of compressing the space-occupying $N \cdot p_\mathrm{I} \times p$ matrices $\mathbf{W}^{(1)}, (\mathbf{W}^{(2)})^\top$, we turn to a theoretical framework *oblivious subspace embedding* (Cohen, 2016, OSE) which can preserve any p-dimensional subspace of $\mathbb{R}^{N\cdot p_\mathrm{I}}$, which, in our case, are the spaces spanned by $\mathbf{W}^{(1)}$ and $(\mathbf{W}^{(2)})^\top$, through a $d \times N \cdot p_\mathrm{I}$ random projection matrix $\mathbf{\Pi}$. The projection $\mathbf{\Pi}\mathbf{W}^{(1)}$ (resp. $\mathbf{\Pi}(\mathbf{W}^{(2)})^\top$ can be considered as a sketch of the expert merging strategy.

We can recognize the limitation of expert merging from the lens of OSE. As per Cohen (2016), the projection dimension

d should be $\mathcal{O}(p \log p/\varepsilon^2)$, where ε is the tolerance level of error; we note in the case of compressing MoE (N will NOT go to infinity), the scale of $p \log p/\varepsilon^2$ will be even larger than $N \cdot p_{\text{I}}$ by simply setting $\varepsilon = 0.05$. The space gain from expert merging is thus usually marginal in this very practical case.

To alleviate the deficiency of OSE above, MC-SMoE leverages the information from training data to allow the merging no longer data-"oblivious" and thus reduce the scale of d. However, despite the overhead cost to first do fine-tuning and the implicit restriction that in inference the testing data have to be i.i.d. as the training data, this therapy will be invalid in the zero-shot setting we mainly consider. We as well empirically verify the above observations on the expert merging strategy through the evaluation in Section 5.4.

4.2. An Extraction Strategy Specific to the MoE Structure.

To handle the MoE modules with limited device memory, we propose to compress the representation of those experts through Wasserstein barycenter. Specifically, we extract an expert E_ω with common pattern from all the experts, and then model the difference between E_ω and E_k by less parameters (we will introduce the difference modeling in section 4.3. We first revisit a view that an MLP can be taken as the ensemble of multiple bottleneck-1 sub-MLPs (Wei et al., 2023a; Wang et al., 2022; Yuan et al., 2022). We rewrite the MLP output as follows:

$$E_k(\mathbf{x}) = \sum_{i=1}^{p_{\text{I}}} \left[\mathbf{W}_{k,\cdot,i}^{(2)} \cdot \sigma \left(\left\langle \mathbf{W}_{k,i,\cdot}^{(1)}, \mathbf{x} \right\rangle + \mathbf{b}_{k,i}^{(1)} \right) \right] + \mathbf{b}_k^{(2)}, \quad (3)$$

where by convention we represent the i-th row (resp. column) in the weight matrix $\mathbf{W}_k^{(1)}$ (resp. $\mathbf{W}_k^{(2)}$) as $\mathbf{W}_{k,i,\cdot}^{(1)}$ (resp. $\mathbf{W}_{k,\cdot,i}^{(2)}$). $\sigma(\cdot)$ is the activation function. The summation implies that an MLP is the ensemble of a few bottleneck-1 sub-MLPs (the sum on the right-hand-side above), which allows a distributional perspective of MLP since the order of the sum does not matter.

Note that various expert network architectures can all be expressed using the structure of multiple bottleneck-1 sub-MLPs. Both the FFN in Mixtral and DeepSeekMoE model use a gated network following Llama (Touvron et al., 2023), whose detailed form can be found in Appendix C.2.

Since $\mathbf{b}_i^{(2)}$ is not involved in the summation in Equation (3), we accordingly quantify the extraction as, after proper permutation, minimizing the squared Frobenius norm of differences between the original weight matrices in each expert and the weight matrices in the barycenter expert:

$$\min_{\substack{\mathbf{W}_\omega^{(1)}, \mathbf{b}_\omega^{(1)}, \mathbf{W}_\omega^{(2)} \\ \mathbf{T}_k \in \mathcal{P}, k \in [N]}} \frac{1}{N} \sum_{k=1}^{N} \Bigg[\left\| \mathbf{T}_k \left[\mathbf{W}_k^{(1)}, \mathbf{b}_k^{(1)} \right] - \left[\mathbf{W}_\omega^{(1)}, \mathbf{b}_\omega^{(1)} \right] \right\|_F^2 + \left\| \mathbf{W}_k^{(2)} \mathbf{T}_k^{\text{T}} - \mathbf{W}_\omega^{(2)} \right\|_F^2 \Bigg], \quad (4)$$

where $\mathcal{P}$ is the class of p_{I}-by-p_{I} permutation matrix and $\mathbf{W}_\omega^{(1)} \in \mathbb{R}^{p_{\text{I}} \times p}, \mathbf{b}_\omega^{(1)} \in \mathbb{R}^{p_{\text{I}}}, \mathbf{W}_\omega^{(2)} \in \mathbb{R}^{p \times p_{\text{I}}}$ are the weight matrices in the barycenter expert E_ω. The introduction of the permutation matrices $\mathbf{T}_k$'s aligns with the distributional perspective of MLPs, that an MLP E_k is equivariant to the row permutation of its design matrix $\mathbf{W}_k = \left[\mathbf{W}_k^{(1)}, \mathbf{b}_k^{(1)}, (\mathbf{W}_k^{(2)})^{\text{T}} \right] \in \mathbb{R}^{p_{\text{I}} \times (2p+1)}$ as the sum's order in Equation (3) is inconsequential. It is worth noting that simultaneously permuting $\mathbf{W}_k^{(1)}$ and $\mathbf{W}_k^{(2)}$ does not affect the expert's output since the permutation matrix is orthogonal.

To solve problem (4), we propose to address the distribution of sub-MLPs within each expert, rather than layer-by-layer. The summation (3) clearly shows the correspondence between the i-th row $\mathbf{W}_{k,i,\cdot}^{(1)}$ of $\mathbf{W}_k^{(1)}$ and the i-th column $\mathbf{W}_{k,\cdot,i}^{(2)}$ of $\mathbf{W}_k^{(2)}$; to obtain the "embedding" of the sub-MLPs for the distributional formulation, we consider the original MLP E_k as a design matrix $\mathbf{W}_k$. We then run the algorithm for free-support Wasserstein barycenter (Cuturi & Doucet, 2014) to obtain the weight matrix $\mathbf{W}_\omega = \left[\mathbf{W}_\omega^{(1)}, \mathbf{b}_\omega^{(1)}, (\mathbf{W}_\omega^{(2)})^{\text{T}} \right]$ for the barycenter expert, with $\mathbf{W}_\omega$ being exactly the solution to the minimization problem (4).

To present the result, we respectively define μ_k's as the uniform distributions defined on the rows of the given $\mathbf{W}_k \in \mathbb{R}^{p_{\text{I}} \times (2p+1)}$, for all $k = 1, 2, \cdots, N$, i.e., $\mu_k = \sum_{i=1}^{p_{\text{I}}} \frac{1}{p_{\text{I}}} \delta_{\mathbf{W}_{k,i,\cdot}}$. Similarly μ_ω is uniformly distributed over the rows of $\mathbf{W}_\omega \in \mathbb{R}^{p_{\text{I}} \times (2p+1)}$, and the notation μ_ω is interchangeable with $\mu_\omega(\mathbf{W}_\omega)$, which highlights the dependence on $\mathbf{W}_\omega$. We further denote the optimal transport matrix (w.r.t. W_2 distance) from μ_k to μ_ω as $\text{OT}(\mu_k, \mu_\omega)$ as the solution of problem (1). We can then give the following proposition (the proof is deferred to Appendix B).

Proposition 4.1. *Consider the solution $\mathbf{W}_\omega$ to the following free-support Wasserstein barycenter problem*

$$\min_{\mathbf{W}_\omega} \frac{1}{N} \sum_{k=1}^{N} W_2^2(\mu_k, \mu_\omega(\mathbf{W}_\omega)). \quad (5)$$

Then $\mathbf{W}_\omega$, along with $\mathbf{T}_k = p_{\text{I}} \cdot \text{OT}\left(\mu_k, \mu_\omega(\mathbf{W}_\omega)\right)$, is the solution to the optimziation problem in Eq. (4).

Remark. Note that all the experts E_k and the barycenter expert E_ω share the same size, i.e., $\mathbf{W}_k, \mathbf{W}_\omega \in \mathbb{R}^{p_{\text{I}} \times (2p+1)}$.

Therefore, the supports for distributions μ_k, μ_ω are of the same size p_{I}. In discrete optimal transport, there is a special property that for two discrete uniform distributions with support of the same size, the optimal transport matrix between the two distributions will be re-scaled as a permutation matrix (Peyre & Cuturi, 2020). The conclusion simplifies the computation since permutation matrix is orthogonal ($\mathbf{T}_k \left(\mathbf{T}_k\right)^{\text{T}} = \mathbf{I}$). The output of the extracted expert $E_\omega(\mathbf{x}) = \mathbf{W}_\omega^{(2)} \sigma\left(\mathbf{W}_\omega^{(1)}\mathbf{x} + \mathbf{b}_\omega^{(1)}\right)$ (adding $\mathbf{b}_k^{(2)}$) is automatically aligned with any expert $E_k(\mathbf{x}), \forall k \in [N]$ without additional transformations.

4.3. "Efficient" Implementation of Unstructured Pruning for Expert Restoration

As the last step, we need to recover the selected expert from the barycenter expert. We consider zeroing out redundant parameters in the residual matrix $\mathbf{T}_k\mathbf{W}_k - \mathbf{W}_\omega$ according to their importance estimation [1] in an entry-wise manner (termed as *unstructured pruning* in literature); equivalently, we will store a sparse matrix $\mathbf{\Delta}_i$ to approximate $\mathbf{T}_k\mathbf{W}_k - \mathbf{W}_\omega$ and then use $\mathbf{\Delta}_k + \mathbf{W}_\omega$ to recover $\mathbf{T}_k\mathbf{W}_k$.

Classical unstructured pruning is not hardware-aware, since sparse matrix multiplication is not well supported on modern hardware (e.g. GPUs) and even slows down the training in wall-clock time (Dao et al., 2022). Instead of turning to structured pruning, in which weight matrices are pruned neuron/column-wisely, our barycenter-based implementation automatically circumvents the hardware issue. The restoring procedure of sparse matrix addition $\mathbf{\Delta}_k + \mathbf{W}_\omega$ is efficient enough on modern hardware, and the resulting dense output will then fully exploit computational devices since the hardware only needs to address regular dense matrix multiplication. We provide more implementation tricks for unstructured pruning in Appendix A.9.

5. Experimental Results

This section starts with our experimental setup, followed by a preliminary evaluation of ResMoE's approximation error against various baselines in Table 1 and its performance in Tables 2 and 3, concluding with an ablation study. All models and methods are implemented in PyTorch. Switch Transformer is fine-tuned on a Tesla V100 32GB GPU, while Mixtral is tested on four such GPUs. Detailed experimental information is available in Appendix A.1.

[1] Popular choices of importance metrics include magnitude (Han et al., 2015b), sensitivity (Sanh et al., 2020; Molchanov et al., 2019), and uncertainty (Zhang et al., 2022).

Table 1. Approximation error of Switch Transformer. The experts are frozen during the fine-tuning stage hence most of the deterministic methods give zero standard deviation. All the numbers are normalized by a factor p_{I} for better reference. UP stands for Unstructured Pruning, and SP stands for Structured Pruning.

	Switch Transformer	Mixtral
UP	34.27±0.00	10.26±0.00
SP	87.00±0.00	27.09±0.00
SVD	56.44±0.00	21.70±0.00
M-SMoE	278.76±0.00	16.73±0.00
MEO	63.25±0.00	15.81±0.00
MLP Fusion	83.45±0.02	27.28±0.01
ResMoE (Ours)	**22.05±0.02**	**6.60±0.01**

5.1. Experiment Setup

Model Backbones. Our evaluation encompasses two primary architectures: the GPT-style causal decoder-only model and the T5-style encoder-decoder model. Specifically, we utilize Mixtral (Jiang et al., 2024) for the decoder-only model, featuring 8 experts per layer across 32 layers. For the encoder-decoder model, we employ the Switch Transformer (Fedus et al., 2022) with a similar expert-layer configuration (switch-base-8) but with 12 encoder layers followed by 12 decoder layers. We fix the router and the experts during the supervised fine-tuning stage, based on the observation that preserving the original LLM's universal world information can enhance their performance (He et al., 2023a; Mukhoti et al., 2023; Dou et al., 2023). This hypothesis is empirically supported by our findings, which demonstrate improved model performance. Additional experiments on DeepSeekMoE (Dai et al., 2024) and switch-base-16 are provided in Appendix A.2 and Appendix A.3.

Compared Methods. We compare our method with different types of baselines. For pruning, we employ both single-shot Unstructured Pruning (Han et al., 2015a; Lee et al., 2019; Tanaka et al., 2020) and Structured Pruning (Li et al., 2023b). We employ truncated SVD following Denton et al. (2014a). For Merge, we employ M-SMoE (the better-performed uncompressed version of MC-SMoE) (Li et al., 2023a) and MEO (He et al., 2023b). We employ Git Re-basin (Ainsworth et al., 2023) for model fusion. We additionally note that Git Re-basin is not initially designed for MoE models, and we dynamically apply it as a merge method to merge the experts according to its applicability to merge multiple models. We also compare our method with MLP Fusion (Wei et al., 2023b), which aims to reduce the intermediate dimension of one expert MLP unit. As our compression rate is set to 25%, we perform different setups to all the methods to make sure they match this setting, detailed in Appendix A.4.

5.2. Preliminary Evaluation of Approximation Error

We calculate the approximation error of each method on the top 8 layers of Switch Transformer and the top 24 layers of Mixtral as a sanity check. The approximation error is defined as the Frobenius norm difference between the original and compressed weight matrices in the expert with details in Appendix A.6. Notably, as we freeze the experts during fine-tuning, most methods show zero standard deviation.

Table 1 shows that ResMoE achieves the lowest Approximation error among all the methods. We use the acronyms UP to represent Unstructured Pruning and SP for Structured Pruning. The results prove that ResMoE manages to retain not only the output of the original model but also the integrity of the internal matrices. Thus, this preliminary experiment successfully validates Proposition 4.1.

5.3. Natural Language Understanding

Experiment Setup. Switch Transformer is fine-tuned then compressed during the inference stage on four Natural Language Understanding (NLU) GLUE tasks, SST-2 (Socher et al., 2013), MRPC (Dolan & Brockett, 2005), CoLA (Warstadt et al., 2019) and MNLI (Williams et al., 2018). All the results are reported with accuracy. Here, all the experiments are conducted using different seeds for three rounds. As not all the layers of Switch Transformer are sparse MoE layers, we perform all the methods at the top 4 encoder's MoE layers and the top 4 decoder's MoE layers.

Results. Table 2 provides the results of Switch Transformer. ResMoE consistently surpasses all baseline methods, underscoring its efficiency in reducing the memory usage in MoE models without the need for further fine-tuning. Unstructured Pruning effectively preserves the original performance, whereas Structured Pruning, applied neuron-wise, exhibits a more pronounced drop. This observation aligns well with our choice of Unstructured Pruning over Structured Pruning. We note a significant difference in performance when applying Pruning and SVD to experts, depending on whether the weights were concatenated or separated. A possible explanation is that Pruning dynamically zeroes out less important weights, retaining crucial ones when expert weights are concatenated, hinting at the benefit of preserving expert-level relationships for model performance. Alongside suboptimal Git Re-basin results, this supports our proposition that previous layer-wise fusion methods are limited. Although most methods manage to preserve the model's performance well in these NLU tasks, they result in a dramatic drop in subsequent Natural Language Generation (NLG) tasks.

5.4. Zero-shot Natural Language Generation

Experiment Setup. Mixtral is tested on WikiText (Language Modelling)(Merity et al., 2016), LAMBADA (Language Modelling)(Paperno et al., 2016), PIQA (Question Answering)(Bisk et al., 2020) and WinoGrande (Common Sense Reasoning)(Sakaguchi et al., 2021). The result of WikiText is given by perplexity, while accuracy metrics are used for LAMBADA, PIQA, and WinoGrande.

As Mixtral's results are tested with zero-shot and fixed weights, this ensures deterministic outcomes for most of the evaluated methods, leading to a standard deviation of 0 for them. However, the Fusion and OT methods, which seek approximate optimization solutions starting from different initial conditions, exhibit variability and therefore have a non-zero standard deviation. All the methods are performed on the top 24 layers, and reduce the parameter counts of the experts to 25% of the original ones.

Results. Table 3 presents the results for the Mixtral model, where ResMoE consistently outperforms all baseline methods, demonstrating its effectiveness in both NLU and NLG tasks. Notably, Structured Pruning results in a substantial performance loss for Mixtral, likely due to its larger hidden dimension (4096), where neuron-wise weight pruning could lead to significant information loss, a situation reminiscent of the MLP Fusion case. It is important to note that Mixtral's experts are initialized through a copy-and-paste method, as opposed to the random Gaussian initialization in Switch Transformer, leading to more uniform weight distributions in Mixtral. This uniformity might contribute to the enhanced performance observed with merge methods in this model. However, the superior performance of ResMoE compared to merge methods further supports our hypothesis about the latter's reduced effectiveness in more generalized scenarios.

5.5. Ablation Studies

The Effectiveness of Wasserstein Barycenter. In ResMoE, we choose to prune the residual matrices between the original experts and the barycenter expert. Here, we study the effectiveness of this choice by conducting the vanilla unstructured pruning without this barycenter expert and using the average experts served as the barycenter. The outcomes in terms of output performance in Table 4 and approximation error in Table 5 clearly demonstrate the beneficial impact of incorporating the barycenter expert.

The Effectiveness of Unstructured Pruning. When compressing the stored residual matrices, we choose unstructured pruning considering that the restore process does not involve matrix multiplication so that the matrix can be stored sparsely. Here, we study the difference between compressing the residual matrices using SVD versus Pruning. Similarly, both the performance outcomes in Table 4 and the approximation error in Table 5 affirm the effectiveness of our choice of unstructured pruning. Notably, vanilla pruning yields great results for the Switch Transformer but underperforms on Mixtral. Conversely, employing SVD to compress

Table 2. Evaluation results of Switch Transformer on four GLUE NLU tasks (measured in accuracy). UP stands for Unstructured Pruning, and SP stands for Structured Pruning. We use 'concat' and 'sep' to denote the concatenate and separate processing of the expert weights. **Bold** results are the best scores under each metric.

	SST-2	MRPC	CoLA	MNLI
Switch Transformer	93.92±0.18	89.54±0.86	82.29±0.28	87.82±0.15
UP (concat)	93.12±0.23	87.75±1.12	81.40±0.48	85.32±0.66
UP (sep)	90.21±0.44	78.92±3.96	79.33±0.80	76.06±5.47
SP (concat)	90.67±0.36	88.72±0.85	79.39±0.42	85.19±0.22
SP (sep)	83.60±1.15	80.88±2.12	76.89±0.93	81.87±1.19
SVD (concat)	92.47±0.04	87.58±1.02	75.90±0.03	85.86±0.07
SVD (sep)	92.59±0.25	87.25±0.93	81.62±0.32	86.04±0.05
M-SMoE	93.31±0.53	87.42±1.06	80.06±0.68	85.72±0.27
Git Re-Basin	84.94±0.86	85.70±0.46	61.90±1.24	83.76±0.84
MEO	92.73±0.39	86.77±0.93	79.99±0.83	85.29±0.37
MLP Fusion	91.86±0.30	88.40±0.59	79.64±0.03	85.72±0.19
ResMoE (Ours)	**93.58±0.07**	**89.21±0.49**	**82.13±0.07**	**86.13±0.09**

Table 3. Zero-shot results of Mixtral. Most of the methods are deterministic based on the model's weights, resulting in a 0 standard deviation. **Bold** results are the best scores under each metric. On the WikiText dataset, where perplexity serves as the evaluation metric. The down-arrow notation (↓) indicates that a lower metric represents better performance.

	WikiText (PPL) ↓	LAMBADA (ACC)	PIQA (ACC)	WinoGrande (ACC)
Mixtral	3.87±0.00	74.05±0.00	82.37±0.00	77.11±0.00
UP	13.03±0.00	36.10±0.00	72.09±0.00	68.59±0.00
SP	13851.63±0.00	0.00±0.00	53.05±0.00	47.91±0.00
SVD	267.94±0.00	16.09±0.00	59.47±0.00	56.99±0.00
M-SMoE	10.45±0.00	58.57±0.00	73.56±0.00	69.61±0.00
Git Re-Basin	9.96±0.00	59.09±0.00	74.70±0.00	69.22±0.00
MEO	8.32±0.00	62.93±0.00	75.84±0.00	70.48±0.00
MLP Fusion	80.06±5.55	5.12±0.49	66.67±0.25	56.80±0.97
ResMoE (Ours)	**5.38±0.04**	**69.44±0.16**	**80.81±0.19**	**74.45±0.23**

the residual matrices is advantageous for Mixtral but less so for Switch Transformer. ResMoE, however, consistently delivers excellent performance across both models.

The Impact of Compression Rate. In the main experiment, we set the compression rate to retain 25% of the parameter counts. Additionally, we explored the impact of adjusting this rate to different levels. Figure 3 provides the results of Mixtral on the LAMBADA dataset. Remarkably, with the compression rate set to 10%, ResMoE manages to achieve results that are not only comparable but even surpass those of baseline methods set at a 30% compression rate, which further demonstrates its superiority.

6. Conclusions and Future Directions

In this paper, we propose ResMoE, a novel one-shot model compression method that reduces the memory usage of Mixture-of-Experts LLMs without any retraining. Instead of directly compressing the experts, we show that the model size can be reduced through pruning the residual between the Wasserstein barycenter and the original experts. We prove the effectiveness of our method by conducting comprehensive experiments on various backbone models, including Switch Transformer with encoder-decoder architecture and the decoder-only Mixtral and DeepSeekMoE. With ResMoE, we prune up to 75% of the parameters with both successful preservation of the original weight matrices and minimal performance loss in the downstream task.

One future direction of this work can be the exploration

Table 4. The comparison of accuracy between vanilla pruning, compressing the residual matrices using SVD and our method. Here UP means Unstructured Pruning, and WB stands for Wasserstein barycenter. **Bold** results are the best scores under each metric.

	Switch Transformer			Mixtral		
	SST-2	MRPC	CoLA	LAMBADA	PIQA	WinoGrande
UP	93.12±0.18	87.75±1.12	81.40±0.20	36.10±0.00	72.09±0.00	68.59±0.00
Avg + UP	92.81±0.42	89.13±0.96	81.62±0.20	67.38±0.00	78.89±0.00	73.95±0.00
WB + SVD	92.85±0.05	88.18±0.48	76.88±0.08	69.28±0.00	80.47±0.00	73.20±0.00
WB + UP	**93.58±0.07**	**89.21±0.49**	**82.13±0.07**	**69.44±0.16**	**80.81±0.19**	**74.45±0.23**

Table 5. The comparison of approximation error. All the numbers are divided by a factor p_{I} for better reference. Here UP means Unstructured Pruning, and WB stands for Wasserstein barycenter.

	Switch Transformer	Mixtral
UP	34.27±0.00	10.26±0.00
WB + SVD	63.09±0.72	22.36±0.02
WB + UP	**22.05±0.02**	**6.60±0.01**

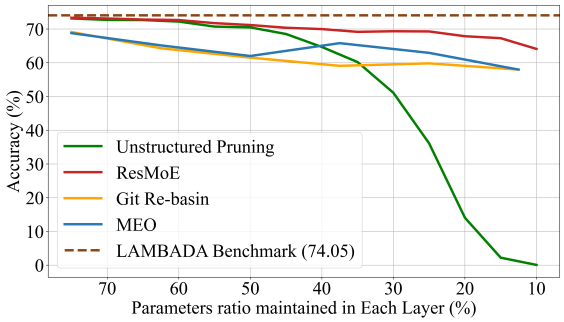

Figure 3. Performance of selected baseline methods on Mixtral w.r.t. various compression rates on the LAMBADA data set. Note that MEO and Git Re-basin can only merge experts into at least one, so that they are not able to reach the 10% compression rate.

of adopting different compression rates for each layer or even each expert (as experimented in Sharma et al. (2023) and MC-SMoE (Li et al., 2023a)), or further combining our method with hardware quantization methods. Additionally, while currently ResMoE is executed without retraining, it can also be applied prior to the supervised fine-tuning stage, potentially easing the hardware requirement for the process.

Broader Impact

As a computational method focused on model compression for Mixture-of-Expert LLMs, our work carries a low ethical risk, with the impact largely dependent on its applications in various downstream tasks. The primary use of our method is to enhance the efficiency of LLMs, which could include applications in natural language processing. Our approach is designed to be data-agnostic, not assuming any specific structure in the input, thereby minimizing the reinforcement of biases. We believe it is unlikely that our method will have negative societal implications, given its focus on improving computational efficiency in language models.

References

Ainsworth, S., Hayase, J., and Srinivasa, S. Git re-basin: Merging models modulo permutation symmetries. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=CQsmMYmlP5T.

Anil, R., Dai, A. M., Firat, O., Johnson, M., Lepikhin, D., Passos, A., Shakeri, S., Taropa, E., Bailey, P., Chen, Z., et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

Bhandare, A., Sripathi, V., Karkada, D., Menon, V., Choi, S., Datta, K., and Saletore, V. Efficient 8-bit quantization of transformer neural machine language translation model, 2019.

Bisk, Y., Zellers, R., Gao, J., Choi, Y., et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020.

Cohen, M. B. Nearly tight oblivious subspace embeddings by trace inequalities. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pp. 278–287. SIAM, 2016.

Cuturi, M. and Doucet, A. Fast computation of wasserstein barycenters. In *International conference on machine learning*, pp. 685–693. PMLR, 2014.

Dai, D., Deng, C., Zhao, C., Xu, R. X., Gao, H., Chen, D., Li, J., Zeng, W., Yu, X., Wu, Y., Xie, Z., Li, Y. K., Huang, P., Luo, F., Ruan, C., Sui, Z., and Liang, W. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models, 2024.

Dao, T., Chen, B., Sohoni, N. S., Desai, A., Poli, M., Grogan, J., Liu, A., Rao, A., Rudra, A., and Ré, C. Monarch: Expressive structured matrices for efficient and accurate training. In *International Conference on Machine Learning*, pp. 4690–4721. PMLR, 2022.

Denton, E. L., Zaremba, W., Bruna, J., LeCun, Y., and Fergus, R. Exploiting linear structure within convolutional networks for efficient evaluation. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014a. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/2afe4567e1bf64d32a5527244d104cea-Paper.pdf.

Denton, E. L., Zaremba, W., Bruna, J., LeCun, Y., and Fergus, R. Exploiting linear structure within convolutional networks for efficient evaluation. *Advances in neural information processing systems*, 27, 2014b.

Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. Llm.int8(): 8-bit matrix multiplication for transformers at scale, 2022.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Dolan, W. B. and Brockett, C. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005. URL https://aclanthology.org/I05-5002.

Dou, S., Zhou, E., Liu, Y., Gao, S., Zhao, J., Shen, W., Zhou, Y., Xi, Z., Wang, X., Fan, X., et al. Loramoe: Revolutionizing mixture of ex-perts for maintaining world knowledge in language model alignment. *arXiv preprint arXiv:2312.09979*, 2023.

Fan, H., Xiong, B., Mangalam, K., Li, Y., Yan, Z., Malik, J., and Feichtenhofer, C. Multiscale vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6824–6835, October 2021.

Fedus, W., Zoph, B., and Shazeer, N. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270, 2022.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Frankle, J., Dziugaite, G. K., Roy, D. M., and Carbin, M. Linear mode connectivity and the lottery ticket hypothesis. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Frantar, E. and Alistarh, D. Sparsegpt: Massive language models can be accurately pruned in one-shot, 2023.

Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. Gptq: Accurate post-training quantization for generative pre-trained transformers, 2023.

Gao, Z.-F., Liu, P., Zhao, W. X., Lu, Z.-Y., and Wen, J.-R. Parameter-efficient mixture-of-experts architecture for pre-trained language models. In Calzolari, N., Huang, C.-R., Kim, H., Pustejovsky, J., Wanner, L., Choi, K.-S., Ryu, P.-M., Chen, H.-H., Donatelli, L., Ji, H., Kurohashi, S., Paggio, P., Xue, N., Kim, S., Hahm, Y., He, Z., Lee, T. K., Santus, E., Bond, F., and Na, S.-H. (eds.), *Proceedings of the 29th International Conference on Computational Linguistics*, pp. 3263–3273, Gyeongju, Republic of Korea, October 2022. International Committee on Computational Linguistics. URL https://aclanthology.org/2022.coling-1.288.

Gou, J., Yu, B., Maybank, S. J., and Tao, D. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129:1789–1819, 2021.

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015a. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/ae0eb3eed39d2bcef4622b2499a05fe6-Paper.pdf.

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015b.

He, G., Chen, J., and Zhu, J. Preserving pre-trained features helps calibrate fine-tuned language models. In *The Eleventh International Conference on Learning Representations*, 2023a. URL https://openreview.net/forum?id=NI7StoWHJPT.

He, S., Fan, R.-Z., Ding, L., Shen, L., Zhou, T., and Tao, D. Merging experts into one: Improving computational efficiency of mixture of experts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 14685–14691, 2023b.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A., Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. Mistral 7b, 2023.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., de las Casas, D., Hanna, E. B., Bressand, F., Lengyel, G., Bour, G., Lample, G., Lavaud, L. R., Saulnier, L., Lachaux, M.-A., Stock, P., Subramanian, S., Yang, S., Antoniak, S., Scao, T. L., Gervet, T., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. Mixtral of experts, 2024.

Lee, N., Ajanthan, T., and Torr, P. SNIP: SINGLE-SHOT NETWORK PRUNING BASED ON CONNECTION SENSITIVITY. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1VZqjAcYX.

Li, P., Zhang, Z., Yadav, P., Sung, Y.-L., Cheng, Y., Bansal, M., and Chen, T. Merge, then compress: Demystify efficient smoe with hints from its routing policy. *arXiv preprint arXiv:2310.01334*, 2023a.

Li, Y., Yu, Y., Zhang, Q., Liang, C., He, P., Chen, W., and Zhao, T. Losparse: Structured compression of large language models based on low-rank and sparse approximation, 2023b.

Li, Z., Wang, Z., Tan, M., Nallapati, R., Bhatia, P., Arnold, A., Xiang, B., and Roth, D. DQ-BART: Efficient sequence-to-sequence model via joint distillation and quantization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 203–211, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.22. URL https://aclanthology.org/2022.acl-short.22.

Liu, Z., Sun, M., Zhou, T., Huang, G., and Darrell, T. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270*, 2018.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Molchanov, P., Mallya, A., Tyree, S., Frosio, I., and Kautz, J. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11264–11272, 2019.

Mukhoti, J., Gal, Y., Torr, P. H. S., and Dokania, P. K. Fine-tuning can cripple your foundation model; preserving features may be the solution, 2023.

OpenAI. Techniques for training large neural networks, 2022. URL https://openai.com/research/techniques-for-training-large-neural-networks.

Paperno, D., Kruszewski, G., Lazaridou, A., Pham, N. Q., Bernardi, R., Pezzelle, S., Baroni, M., Boleda, G., and Fernández, R. The LAMBADA dataset: Word prediction requiring a broad discourse context. In Erk, K. and Smith, N. A. (eds.), *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1525–1534, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1144. URL https://aclanthology.org/P16-1144.

Peyre, G. and Cuturi, M. Computational optimal transport, 2020.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer, 2023.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Sanh, V., Wolf, T., and Rush, A. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in Neural Information Processing Systems*, 33:20378–20389, 2020.

Sharma, P., Ash, J. T., and Misra, D. The truth is in there: Improving reasoning in language models with layer-selective rank reduction, 2023.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, 2017.

Singh, S. P. and Jaggi, M. Model fusion via optimal transport. *Advances in Neural Information Processing Systems*, 33:22045–22055, 2020.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Tanaka, H., Kunin, D., Yamins, D. L., and Ganguli, S. Pruning neural networks without any data by iteratively conserving synaptic flow. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6377–6389. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/46a4378f835dc8040c8057beb6a2da52-Paper.pdf.

Tao, C., Hou, L., Zhang, W., Shang, L., Jiang, X., Liu, Q., Luo, P., and Wong, N. Compression of generative pre-trained language models via quantization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4821–4836, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.331. URL https://aclanthology.org/2022.acl-long.331.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. Llama 2: Open foundation and fine-tuned chat models, 2023.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wang, B., Ren, Y., Shang, L., Jiang, X., and Liu, Q. Exploring extreme parameter compression for pre-trained language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=RftryyYyjiG.

Wang, C., Zhang, G., and Grosse, R. Picking winning tickets before training by preserving gradient flow. *arXiv preprint arXiv:2002.07376*, 2020.

Wang, W., Xie, E., Li, X., Fan, D.-P., Song, K., Liang, D., Lu, T., Luo, P., and Shao, L. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 568–578, October 2021.

Warstadt, A., Singh, A., and Bowman, S. R. Neural network acceptability judgments. *Transactions of the Association*

for Computational Linguistics, 7:625–641, 2019. doi: 10.1162/tacl_a_00290. URL https://aclanthology.org/Q19-1040.

Wei, T., Guo, Z., Chen, Y., and He, J. NTK-approximating MLP fusion for efficient language model fine-tuning. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 36821–36838. PMLR, 23–29 Jul 2023a. URL https://proceedings.mlr.press/v202/wei23b.html.

Wei, T., Guo, Z., Chen, Y., and He, J. Ntk-approximating mlp fusion for efficient language model fine-tuning. In *International Conference on Machine Learning*, pp. 36821–36838. PMLR, 2023b.

Williams, A., Nangia, N., and Bowman, S. A broad-coverage challenge corpus for sentence understanding through inference. In Walker, M., Ji, H., and Stent, A. (eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1101. URL https://aclanthology.org/N18-1101.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Xia, M., Gao, T., Zeng, Z., and Chen, D. Sheared llama: Accelerating language model pre-training via structured pruning, 2023.

Xue, F., He, X., Ren, X., Lou, Y., and You, Y. One student knows all experts know: From sparse to dense, 2022.

Yuan, B., Wolfe, C. R., Dun, C., Tang, Y., Kyrillidis, A., and Jermaine, C. Distributed learning of fully connected neural networks using independent subnet training. *Proceedings of the VLDB Endowment*, 15(8):1581–1590, 2022.

Zhang, Q., Zuo, S., Liang, C., Bukharin, A., He, P., Chen, W., and Zhao, T. Platon: Pruning large transformer models with upper confidence bound of weight importance. In *International Conference on Machine Learning*, pp. 26809–26823. PMLR, 2022.

A. Details for Experiments

The models are implemented through the 'transformers' package developed by Wolf et al. (2020). When using Switch Transformer for the GLUE classification tasks, we add a classification head according to the setting of T5.

A.1. Details of Experiment Settings

In the zero-shot context, we adhere to the standard practice for zero-shot LLM evaluation. This involves assessing the pre-trained LLM on various downstream tasks without any additional fine-tuning. In the fine-tuning scenario, we employ the typical procedure of initially fine-tuning the pre-trained model on downstream tasks, followed by evaluation during the inference stage. Our approach is concentrated on enhancing the inference stage, ensuring that after the implementation of our methods, no further fine-tuning is required, and the compressed model will be ready-to-use.

Table 6. Fine-tuning hyper-parameters setting for Switch Transformer.

	Value
Optimizer	AdamW
Adam ϵ	1e-08
Adam β	(0.9, 0.98)
warm-up steps	8
weight decay	0.01

For Mixtral, we adhere to the model's configuration card, as no hyperparameter tuning is required for zero-shot tasks. For Switch Transformer, we employ AdamW optimizer with a linear warm-up step count of 8. We explore the hyper-parameters settings during the supervised fine-tuning stage. For Switch Transformer, the learning rate is searched in the range of {1e-4,2e-4,3e-4,5e-4,1e-3}, the batch size within the range of {16,32,64}, and the training epoch within the range of {3,5,10,15,20}. The details of the AdamW optimizer which is fixed for all datasets are given in table 6.

To ensure comparability across methods, we standardize the parameter count reduction for the experts to around 75%, which means 25% of the parameters will be retained. All the methods are performed at the top 24 layers of Mixtral, and the top 8 MoE layers of Switch Transformer. All the methods are applied during the inference stage.

A.2. Experiment Results of DeepSeekMoE

DeepSeekMoE (Dai et al., 2024) features more fine-grained experts compared to alternative structures, with each expert sized at just 8.7M, markedly smaller than Mixtral's 176.2M. It introduces a unique, independent shared expert atop each MoE layer, aiming to encapsulate universal information across layers. This approach is inspired by observations from Mixtral's router analysis, which indicates a roughly equal routing probability for each expert during the inference stage, suggesting the presence of universal knowledge within each expert. Consequently, an additional shared expert is integrated, anticipated to store universal information and thereby enhance the diversity of the remaining experts. In our experiments with DeepSeekMoE, we exclude this shared expert, considering its anticipated information content significantly differs from that of the non-shared experts.

Table 7 provides the results for DeepSeekMoE. Note that the results for LAMBADA dataset are not included here, since DeepSeekMoE produces extremely bad results for this dataset (0.04% Accuracy). Even though the experts of DeepSeekMoE are also initialized through copy-and-paste, the existence of the shared expert distinguishes those experts in the MoE layers from each other, leading to the poor performance of the merge methods, which aligns with our proposition that merge methods will potentially impair the generalization ability of the original model.

A.3. Experiment Results of Switch Transformer (switch-base-16)

Following the same fine-tuning settings as those used for switch-base-8, detailed in Appendix A.1, we limited our testing of switch-base-16 to the MRPC dataset due to the constraints of time-intensive supervised fine-tuning. Despite this limitation, Table 8 allows us to draw a similar conclusion as with switch-base-8, that ResMoE consistently demonstrates impressive results, affirming its efficacy in maintaining model accuracy. Additionally, akin to the observations from switch-base-8, we

Table 7. Zero-shot results of DeepSeekMoE. For Pruning, we choose Unstructured Pruning over Structured Pruning based on the observations from the previous experiments that Unstructured Pruning usually has better results on NLG tasks.

	WikiText (PPL)↓	PIQA (ACC)	WinoGrande (ACC)
DeepSeekMoE	6.51	78.84	68.75
Pruning	10.46	73.12	62.83
SVD	26.93	63.06	57.46
M-SMoE	34.76	62.79	54.22
MEO	33.94	62.35	54.14
ResMoE (Ours)	**10.39±0.10**	**73.39±0.01**	**64.35±0.01**

Table 8. Evaluation Results of Switch Transformer (switch-base-16), with 16 experts per layer. UP stands for Unstructured Pruning, and SP stands for Structured Pruning. We use 'concat' to denote concatenate and 'sep' to denote separate.

	MRPC
Switch Transformer	90.03±0.45
UP (concat)	89.47±0.01
UP (sep)	88.48±0.75
SP (concat)	88.40±0.94
SP (sep)	87.34±0.46
SVD (concat)	88.48±0.62
SVD (sep)	88.48±0.62
M-SMoE	88.89±0.59
MEO	88.51±0.93
MLP Fusion	87.91±0.90
ResMoE (Ours)	**89.62±0.01**

notice that the choice between pruning or applying SVD to the weights, whether concatenated or separated, significantly influences the outcomes. This consistency across different model scales reinforces the impact of these compression techniques on the model's performance, highlighting the nuanced balance between efficiency and accuracy in model optimization.

A.4. Compression Setting For Each Method

In order to match our setting of the 75% compression rate, each method has to be tailored differently. For Pruning, we mask 75% weights units with the lowest L1-norm within each expert. For SVD, the details of calculating the rate of the parameters can be referred to in Appendix A.5. For M-SMoE, Git Re-basin, and MEO, we reduce the expert count of each MoE layer from 8 to 2. For MLP Fusion, we reduce the intermediate dimension to 25%. For ResMoE, we mask 75% weights units with the lowest L1-norm within each residual matrix. We do not count the overhead storage of the barycenter experts here since we aim to prove the effectiveness of our algorithm, and as the number of experts grows, the redundancy of this overhead will diminish. We provide additional experiments performed at switch-base-16 and DeepSeekMoE (64 experts per layer) (Dai et al., 2024) as a support, which can be found in Appendix A.2 and Appendix A.3.

A.5. The Parameter Count of SVD

For SVD, to make the the parameters retained for each expert equal, for Switch Transformer we have:

$$p_{\mathrm{I}} \times k + k + k \times 2p \approx k \times (p_{\mathrm{I}} + 2p)$$
$$s \times p_{\mathrm{I}} \times 2p = 2spp_{\mathrm{I}},$$

where s is the parameter rate we retain (25% here), and k is the number of top-k singular values in SVD. For Switch Transformer, we have $p_{\text{I}} = 4p$, so $k = \frac{1}{3} s p_{\text{I}}$.

For Mixtral:

$$p_{\text{I}} \times k + k + k \times 3p \approx k \times (p_{\text{I}} + 3p)$$
$$s \times p_{\text{I}} \times 3p = 3spp_{\text{I}},$$

here we have $p_{\text{I}} = 3.5p$, so $k = \frac{6}{13} s p_{\text{I}}$.

For DeepSeekMoE:

$$p_{\text{I}} \times k + k + k \times 3p \approx k \times (p_{\text{I}} + 3p)$$
$$s \times p_{\text{I}} \times 3p = 3spp_{\text{I}},$$

here we have $p_{\text{I}} = \frac{131}{256} p$, so $k = \frac{768}{899} s p_{\text{I}}$.

A.6. The Defination of Approximation Error

Take Switch Transformer for example, since there is no bias in this model, the approximation error ϵ for one layer is defined as:

$$\epsilon = \frac{1}{N} \sum_{k=1}^{N} \left[\left\| \mathbf{T}_k \mathbf{W}_k^{(1)} - \hat{\mathbf{W}}_k^{(1)} \right\|_F^2 + \left\| \mathbf{W}_k^{(2)} \mathbf{T}_k^T - \hat{\mathbf{W}}_k^{(2)} \right\|_F^2 \right],$$

where $\hat{\mathbf{W}}_k$ denotes the matrix post-application of each method. For ResMoE, $\hat{\mathbf{W}} = \mathbf{W}_\omega + \mathbf{\Delta}_k$, with $\mathbf{\Delta}_k$ being the sparse residual matrix derived from each layer, and $\mathbf{W}_\omega$ as the Wasserstein barycenter matrix. For merge methods, $\hat{\mathbf{W}} = \mathbf{W}_\omega$, where $\mathbf{W}_\omega$ is the merged center of each group. For methods not involving permutation operations, we set $\mathbf{T}_k = \mathbf{I}$. Specifically for MLP fusion which reduces the MLP's weight matrix size, it still allows approximation error computation, as detailed in Appendix A.7.

A.7. Evaluation of Approximation Error for MLP Fusion

We first reformulate MLP fusion (Wei et al., 2023b) with the notations in this paper. In computing the fused MLP for expert k, we obtain the centroid weight/bias $\widetilde{\mathbf{W}}_k = \left[\widetilde{\mathbf{W}}_k^{(1)}, \tilde{\mathbf{b}}_k^{(1)}, (\widetilde{\mathbf{W}}_k^{(2)})^T \right] \in \mathbb{R}^{c \times (2p+1)}$, as well as the one-hot clustering matrix $\mathbf{C}_k \in \mathbb{R}^{c \times p_{\text{I}}}$ indicating how the p_{I} neurons are partitioned into c clusters. MLP fusion then proposes to compute

$$\widetilde{\mathbf{W}}_k^{(2)} \left(\mathbf{C}_k \mathbf{C}_k^T \right) \sigma \left(\widetilde{\mathbf{W}}_k^{(1)} \mathbf{x} + \tilde{\mathbf{b}}_k^{(1)} \right) + \mathbf{b}_k^{(2)}$$

as the approximation to the original expert k.

Wei et al. (2023b) suggest the expression above is equivalent to replacing $\mathbf{W}_k = \left[\mathbf{W}_k^{(1)}, \mathbf{b}_k^{(1)}, (\mathbf{W}_k^{(2)})^T \right] \in \mathbb{R}^{p_{\text{I}} \times (2p+1)}$ by $\mathbf{C}_k^T \widetilde{\mathbf{W}}_k$, considering

$$\widetilde{\mathbf{W}}_k^{(2)} \left(\mathbf{C}_k \mathbf{C}_k^T \right) \sigma \left(\widetilde{\mathbf{W}}_k^{(1)} \mathbf{x} + \tilde{\mathbf{b}}_k^{(1)} \right) + \mathbf{b}_k^{(2)} = \left(\widetilde{\mathbf{W}}_k^{(2)} \mathbf{C}_k \right) \sigma \left(\mathbf{C}_k^T \widetilde{\mathbf{W}}_k^{(1)} \mathbf{x} + \mathbf{C}_k^T \tilde{\mathbf{b}}_k^{(1)} \right) + \mathbf{b}_k^{(2)}.$$

We can thus calculate $\left\| \mathbf{W}_k - \mathbf{C}_k^T \widetilde{\mathbf{W}}_k \right\|_F^2$ as the approximation error on expert k for MLP fusion.

A.8. Details of the Datasets

We provide the details of the datasets we used in the experiment along with their license here.

- PIQA(Bisk et al., 2020): PIQA, or Physical Interaction Question Answering, is a benchmark dataset that assesses AI systems' commonsense reasoning abilities regarding physical knowledge. It challenges models with multiple-choice questions related to everyday physical interactions, testing their understanding of object manipulation and functionality in real-world scenarios. While humans perform well on PIQA, it presents a significant challenge to AI models, making it crucial for advancing AI research, especially in robotics and conversational AI. PIQA is licensed under Academic Free License v3.0.

- WikiText(Merity et al., 2016): WikiText-103 is a widely-used dataset in Natural Language Processing, ideal for language modeling and text generation tasks. Derived from verified Wikipedia articles, it offers over 100 million tokens of well-structured text, preserving original formatting. Its extensive vocabulary and varied syntax make it a valuable resource for training advanced language models. WikiText-103 serves as a crucial benchmark for evaluating language models' performance in handling real-world textual data, with a license of CC BY-SA 3.0.

- WinoGrande(Sakaguchi et al., 2021): Winogrande, an extension of the Winograd Schema Challenge, consists of ambiguous sentence pairs requiring deep language understanding and commonsense reasoning to resolve. It aims to overcome limitations in previous datasets and assesses AI models' ability to comprehend nuanced language and context, making it vital for advancing natural language understanding. Winogrande is licensed under CC-BY.

- LAMBADA(Paperno et al., 2016): LAMBADA is a challenging benchmark designed for evaluating computational models' language understanding, focusing on predicting the final word in a passage. It requires models to grasp broad context and long-range dependencies within text passages. LAMBADA pushes the boundaries of language models, particularly in handling complex, context-dependent linguistic phenomena, making it a valuable tool for advancing natural language processing. It is licensed under CC BY 4.0.

- SST-2(Socher et al., 2013): SST-2, the Stanford Sentiment Treebank version 2, is a popular dataset for sentiment analysis. It contains movie review sentences labeled as positive or negative, excluding neutral sentences, providing a binary classification task. This dataset is notable for its fine-grained annotation, as it includes sentiment labels for every subphase within the sentence parse trees. SST-2 is widely used for training and evaluating models on sentiment analysis, testing their ability to understand nuanced emotional tones in text, with the license of CC0: Public Domain.

- MRPC(Dolan & Brockett, 2005): The Microsoft Research Paraphrase Corpus (MRPC) evaluates models on paraphrase identification by using sentence pairs from online news sources. MRPC is a part of the GLUE benchmark and is valuable for assessing a model's ability to understand and compare semantic content in sentences, especially in semantic analysis tasks. The license of MRPC is unknown.

- CoLA(Warstadt et al., 2019): The Corpus of Linguistic Acceptability (CoLA) assesses models' linguistic acceptability judgment. It distinguishes between grammatically acceptable and unacceptable sentences, emphasizing the importance of grammatical understanding in language comprehension and model evaluation. The license for CoLA is not specified.

- MNLI(Warstadt et al., 2019): The Multi-Genre Natural Language Inference (MNLI) dataset is a diverse corpus for natural language understanding tasks, focusing on textual entailment. It includes pairs of sentences and challenges models to determine whether the second sentence entails, contradicts, or remains neutral to the first sentence. MNLI's wide range of genres and diverse content makes it a robust benchmark for evaluating models in natural language inference tasks. Most of the data are under the OANC's license, with the other falling under several permissive licenses, a Creative Commons Share-Alike 3.0 Unported License, and Creative Commons Attribution 3.0 Unported Licenses.

A.9. Implementation Trick

In our application of unstructured pruning with ResMoE, a key consideration is its impact on memory storage. The Pytorch version we use supports only the Coordinate format (COO) for sparse matrices, which necessitates storing indices as int64. For instance, in an MLP of Mixtral, the original memory footprint is 672MB. However, a version pruned to 75% sparsity consumes more memory, around 840MB, with 672MB used just for storing indices. If the indices could be stored as int16, the memory requirement for the indices would be reduced to 168MB, thus the memory of the entire MLP unit would be significantly reduced to 336MB. Furthermore, if the index is stored in the format of CSR instead of COO, the memory size can be reduced to 252MB. Although addressing this limitation falls outside our current scope, we aim to explore solutions to this challenge in future work.

	Memory Usage (MB)
Mixtral	5376
UP	2016
SP	1344
SVD	1344
M-SMoE	1344
Git Re-basin	1344
MEO	1344
MLP Fusion	1344
ResMoE (WB+UP)	2688
ResMoE (WB+SVD)	2016

Table 9. Memory Usage of One MoE Layer in Mixtral

	Memory Usage (MB)
DeepSeekMoE	2112
UP	1056
SP	528
SVD	528
M-SMoE	528
Git Re-basin	528
MEO	528
MLP Fusion	528
ResMoE (WB+UP)	1089
ResMoE (WB+SVD)	561

Table 10. Memory Usage of One MoE Layer in DeepSeekMoE

The reason we "claim ResMoE is hardware friendly" is that the sparse matrices made by traditional unstructured pruning cannot be stored as real sparse matrices since the **matrix multiplication** operation is not well-supported on modern hardware. However, in the process of ResMoE, those residual sparse matrices will not be involved in any **multiplication** process, instead just matrix sum operation, which is actually no harm to the modern hardware.

On the other hand, even though ResMoE currently adopts unstructured pruning, other compression methods can also be adopted. As in Section 5.5, we choose SVD as the compression method for the ablation study, which will be able to directly reduce memory usage since SVD will reduce the size of each matrix. We remark that even though utilizing SVD here leads to slightly worse results than unstructured pruning, it still performs better compared to the baseline methods (See Table 4 of our paper). Also, when the number of experts goes up, the overhead introduced by the center expert diminishes.

Based on such settings, we provide the memory information on Mixtral (8 experts per layer) and DeepSeekMoE (64 experts per layer) with the overhead center expert included. We remark that even though utilizing SVD in ResMoE leads to slightly worse results than unstructured pruning, it still performs better compared to the baseline methods (See Table 4 of our paper). Also, when the number of experts goes up, the overhead memory introduced by the center expert diminishes.

In addition to the memory information, we also provide the evaluation on runtime. The runtime is obtained by testing on 2 A100 GPUs on the WinoGrande Dataset with a batch size of 64. It is worth noting that the runtime of the merged methods is even slower compared to the original Mixtral model. This is likely due to the code we referred from (Li et al., 2023a), which only creates references from the experts that are merged but does not exactly reduce the number of experts in the model. Note that the sparse matrices induced by unstructured pruning are stored as normal matrices instead of sparse matrices here.

A.10. Pseudocode code of ResMoE

	Runtime (s)
Mixtral	39.44±0.30
UP	39.01±0.21
SP	37.15±0.22
M-SMoE	49.19±0.12
Git Re-basin	48.53±0.09
MEO	49.51±0.18
MLP Fusion	38.53±0.26
ResMoE	38.85±0.28

Table 11. Runtime of Mixtral on WinoGrade

Algorithm 1 ResMoE

Input: experts weights $E_1, \ldots, E_n$, sparsity ratio r
Output: center expert C, pruned residuals $R_1, \ldots, R_n$
// Step 1: Calculate the center expert using free-support Wasserstein barycenter
$C \leftarrow$ free_support_wasserstein_barycenter$(E_1, \ldots, E_n)$
// Step 2: Calculate the residual matrices
for $i = 1$ **to** n **do**
  $R_i \leftarrow E_i - C$
end for
// Step 3: Apply unstructured pruning to residual matrices using L1-norm
for $i = 1$ **to** n **do**
  $R_i \leftarrow$ unstructured_pruning_l1(R_i, r)
end for
return $C, R_1, \ldots, R_n$

A.11. Additional Results

We conduct further experiments on the dense model for reference. We also add further experiments on BoolQ (Reading comprehension) and GSM8K (Math).

For Switch Transformer, we compare it with t5-base. We could see that ResMoE overall performs better than the t5-base model.

For Mixtral, we compare it with Mistral 7B. All the results but WikiText are represented with accuracy, with the latter measured through perplexity. It is worth mentioning that Mixtral is under the setting of zero-shot, which would lead to a higher performance drop compared to the supervised fine-tuned (SFT) Switch Transformer. However, even the 75% sparsity setting of ResMoE in our paper shows no better results than Mistral 7B, setting the sparsity to 50% (which is a more

Algorithm 2 Inference

Input: input x, center expert C, pruned residuals $R_1, \ldots, R_n$, selected experts subscript set S
Output: inference result y
// Step 1: Reconstruct and dynamically load the compressed experts
$E_S \leftarrow \emptyset$
for $R_i \in R_S$ **do**
  $E_i \leftarrow C + R_i$
  $E_S \leftarrow E_S \cup E_i$
end for
// Step 2: Perform inference using the recovered experts
$y \leftarrow$ perform_inference(x, E_S)
return y

	SST-2	MRPC	CoLA	MNLI
t5-base (dense)	93.46 ± 0.31	89.22 ± 0.14	81.97 ± 0.22	85.77 ± 0.3
switch-base-8	93.92 ± 0.18	89.54 ± 0.86	82.29 ± 0.28	87.82 ± 0.15
ResMoE (w/ 75% sparsity)	93.58 ± 0.07	89.21 ± 0.49	82.13 ± 0.07	86.13 ± 0.09

Table 12. Further Experiments on Switch Transformer

	WinoGrande	WikiText (PPL)	PIQA	LAMBADA	BoolQ	GSM8K
Mistral (dense)	75.06 ± 0.00	5.25 ± 0.00	80.74 ± 0.00	70.35 ± 0.00	68.62 ± 0.00	38.13 ± 0.00
Mixtral	77.11 ± 0.00	3.87 ± 0.00	82.37 ± 0.00	74.05 ± 0.00	73.82 ± 0.00	58.98 ± 0.00
ResMoE (75%)	74.45 ± 0.23	5.38 ± 0.04	80.81 ± 0.19	69.44 ± 0.16	66.03 ± 0.11	35.69 ± 0.21
ResMoE (50%)	76.03 ± 0.30	4.29 ± 0.01	82.51 ± 0.11	71.43 ± 0.18	71.66 ± 0.22	54.66 ± 0.25

Table 13. Further Experiments on Mixtral

common ratio for unstructured pruning (Frantar & Alistarh, 2023)) can lead to a better performance than Mistral 7B.

We would also like to refer to some other results from other papers that are worse than ours under the same setting. For example, SparseGPT's (Frantar & Alistarh, 2023) performance on WikiText according to Figure 5 drops from 7.5 to 17.5 (perplexity), and Sheared-LLaMA's (Xia et al., 2023) performance drops even more according to Table 2, from 69.3 (7B) to 64.2 (2.7B) / 57.9 (1.3B) on WinoGrande (under our setting, the similar parameter size should be around 1.75B).

B. Proof of Proposition 4.1

For the reader's convenience, we recall Proposition 4.1 as follows.

Proposition 4.1. *Consider the solution $\mathbf{W}_\omega$ to the following free-support Wasserstein barycenter problem*

$$\min_{\mathbf{W}_\omega} \frac{1}{N} \sum_{k=1}^{N} W_2^2(\mu_k, \mu_\omega(\mathbf{W}_\omega)). \tag{5}$$

Then $\mathbf{W}_\omega$, along with $\mathbf{T}_k = p_{\text{I}} \cdot \text{OT}\left(\mu_k, \mu_\omega(\mathbf{W}_\omega)\right)$, is the solution to the optimziation problem in Eq. (4).

Proof. We recall μ_i, μ_ω are uniformly distributed over the p_{I} rows of $\mathbf{W}_i$ and $\mathbf{W}_\omega$, respectively. $\text{OT}\left(\mu_i, \mu_\omega(\mathbf{W}_\omega)\right)$ is the optimal transport matrix $\mathbf{M}$ from μ_i to μ_ω of the following problem:

$$\text{OT}(\mu, \nu) := \underset{\mathbf{M} \in U(\alpha, \beta)}{\text{argmin}} \left\langle \mathbf{M}, \mathbf{C} \right\rangle, \tag{6}$$

where $\mathbf{C} = \left[\|\mathbf{W}_{k_i} - \mathbf{W}_{\omega_j})\|^2\right]_{ij} \in \mathbb{R}^{p_I \times p_I}, U(\frac{\mathbf{1}_{p_I}}{p_I}, \frac{\mathbf{1}_{p_I}}{p_I}) := \{\mathbf{M} \in \mathbb{R}_+^{p_I \times p_I} \mid \mathbf{M}\mathbf{1}_{p_I} = \frac{\mathbf{1}_{p_I}}{p_I}, \mathbf{M}^T\mathbf{1}_{p_I} = \frac{\mathbf{1}_{p_I}}{p_I}\}$.

We first relate the transport matrix to the permutation matrices $\mathbf{T}_k$'s. Peyre & Cuturi (2020, Proposition 2.1) shows the optimal solution to problem (6) is exactly a permutation matrix, up to a constant factor p_{I}. Now straightforwardly, problem (4) can be rewritten as:

$$\min_{\substack{\mathbf{W}_\omega \\ \mathbf{T}_k \in P, k \in [N]}} \frac{1}{N} \sum_{k=1}^{N} \left[\|\mathbf{T}_k \mathbf{W}_k - \mathbf{W}_\omega\|_F^2 \right], \tag{7}$$

where $\mathbf{W}_k = [\mathbf{W}_k^{(1)}, \mathbf{b}_k^{(1)}, (\mathbf{W}_k^{(2)})^T] \in \mathbb{R}^{p_I \times (2p+1)}$, and $\mathbf{W}_\omega = [\mathbf{W}_\omega^{(1)}, \mathbf{b}_\omega^{(1)}, (\mathbf{W}_\omega^{(2)})^T] \in \mathbb{R}^{p_I \times (2p+1)}$.

We denote the objective function in problem (7) as $f(\mathbf{W}_\omega; \{\mathbf{T}_k\}_{k=1}^N) = \sum_{k=1}^N \frac{1}{N}[\|\mathbf{T}_k \mathbf{W}_k - \mathbf{W}_\omega\|_F^2]$, and take $\mathbf{W}_\omega^*$ as the optimal solution to the Wasserstein barycenter problem (5). For the given $\mathbf{W}_\omega^*$, we further denote $\mathbf{T}_k^* := \underset{\mathbf{T}_k}{\text{argmin}} f(\mathbf{W}_w^*; \mathbf{T}_k), \forall k \in [N]$. The rest of the proof is to show $f(\mathbf{W}_\omega^*; \{\mathbf{T}_k^*\}_{k=1}^N) =$ Equation (4).

① We start with the first side: Equation (4) $\leq f(\mathbf{W}_\omega^*; \{\mathbf{T}_k^*\}_{k=1}^N)$. We indeed immediately have:

$$(4) = \min_{\mathbf{W}_\omega, \mathbf{T}_k} f(\mathbf{W}_\omega; \{\mathbf{T}_k\}_{k=1}^N) \leq f(\mathbf{W}_\omega^*; \{\mathbf{T}_k^*\}_{k=1}^N),$$

due to the definition of min in Equation (4).

② For the other direction, we first show the barycenter loss (5) $\leq$ (4). Through the definition of W_2 distance, we have

$$
\begin{aligned}
& W_2^2(\mu_i, \mu_\omega(\mathbf{W}_\omega)) \leq \|\mathbf{T}_k\mathbf{W}_k - \mathbf{W}_\omega\|_F^2, \ \forall \mathbf{T}_k, \mathbf{W}_\omega \\
\Rightarrow & \frac{1}{N}\sum_{k=1}^N W_2^2(\mu_i, \mu_\omega(\mathbf{W}_\omega)) \leq \frac{1}{N}\sum_{k=1}^N \|\mathbf{T}_k\mathbf{W}_k - \mathbf{W}_\omega\|_F^2, \ \forall \mathbf{T}_k, \mathbf{W}_\omega \\
\Rightarrow & \frac{1}{N}\sum_{k=1}^N W_2^2(\mu_i, \mu_\omega(\mathbf{W}_\omega)) \leq \frac{1}{N}\sum_{k=1}^N \min_{\mathbf{T}_k} \|\mathbf{T}_k\mathbf{W}_k - \mathbf{W}_\omega\|_F^2, \ \forall \mathbf{W}_\omega.
\end{aligned}
$$

The inequality will still hold when we minimize the two sides both over $\mathbf{W}_\omega$:

$$
(5) = \min_{\mathbf{W}_\omega} \frac{1}{N}\sum_{k=1}^N W_2^2(\mu_i, \mu_\omega(\mathbf{W}_\omega)) \leq \min_{\mathbf{W}_\omega} \frac{1}{N}\sum_{k=1}^N \min_{\mathbf{T}_k} \|\mathbf{T}_k\mathbf{W}_k - \mathbf{W}_\omega\|_F^2 = (4).
$$

To close the proof, it suffices to show that $f(\mathbf{W}_\omega^*; \mathbf{T}_k^*) = (5)$. We show the equivalence as follows:

$$
\begin{aligned}
f(\mathbf{W}_\omega^*; \mathbf{T}_k^*) &= \frac{1}{N}\sum_{k=1}^N \|\mathbf{T}_k^*\mathbf{W}_k - \mathbf{W}_\omega^*\|_F^2 = \frac{1}{N}\sum_{k=1}^N \min_{\mathbf{T}_k}[\|\mathbf{T}_k\mathbf{W}_k - \mathbf{W}_\omega^*\|_F^2] \\
&= \frac{1}{N}\sum_{k=1}^N W_2^2(\mu_i, \mu_\omega(\mathbf{W}_\omega^*)),
\end{aligned}
$$

where the last equation holds again thanks to Peyre & Cuturi (2020, Proposition 2.1). Using the fact that W_ω^* is the optimal solution to Wasserstein barycenter problem (5), we finally attain

$$
\begin{aligned}
f(\mathbf{W}_\omega^*; \mathbf{T}_k^*) &= \frac{1}{N}\sum_{k=1}^N W_2^2(\mu_i, \mu_\omega(\mathbf{W}_\omega^*)) = \min_{\mathbf{W}_\omega} \frac{1}{N}\sum_{i=1}^N W_2^2(\mu_i, \mu_\omega(\mathbf{W}_\omega)) \\
&= (5),
\end{aligned}
$$

which completes the proof. ♢

C. Useful Facts and Background Knowledge

C.1. Comparisons between ResMoE and Previous Baselines

In Figure 4, we visually compare our proposed ResMoE with previous baselines. Expert merging techniques consolidate multiple experts into a single entity, while pruning strategies involve the direct removal of connections within individual experts. Low-rank methods, on the other hand, aim to reduce the matrix size within each expert as a means of achieving compression. Our ResMoE first obtains the common barycenter expert and subsequently compresses the residuals between each expert and the barycenter expert, which can be effectively and efficiently used for restoring in the inference stage.

C.2. The MLP Form for Mixtral and DeepSeekMoE

The output of an MLP in Mixtral and DeepseekMoE can be rewritten into:

$$
\begin{aligned}
E_k(\mathbf{x}) &= \mathbf{W}_k^{(2)} \cdot \mathrm{SwiGLU}(\mathbf{x}) + \mathbf{b}_k^{(2)} \\
&= \mathbf{W}_k^{(2)} \cdot \left[\sigma\left(\mathbf{W}_k^{(1)} \cdot \mathbf{x} + \mathbf{b}_k^{(1)}\right) \odot \left(\mathbf{W}_k^{(3)} \cdot \mathbf{x} + \mathbf{b}_k^{(3)}\right)\right] + \mathbf{b}_k^{(2)} \\
&= \sum_{i=1}^{p_I} \mathbf{W}_{k,\cdot,i}^{(2)} \cdot \left[\sigma\left(\left\langle \mathbf{W}_{k,i,\cdot}^{(1)}, \mathbf{x}\right\rangle + \mathbf{b}_{k,i}^{(1)}\right) \cdot \left(\left\langle \mathbf{W}_{k,i,\cdot}^{(3)}, \mathbf{x}\right\rangle + \mathbf{b}_{k,i}^{(3)}\right)\right] + \mathbf{b}_k^{(2)},
\end{aligned}
$$

where $\sigma(\mathbf{x}) = \mathrm{Swish}_\beta(\mathbf{x}) = \mathbf{x}\sigma(\beta\mathbf{x}) = \dfrac{\mathbf{x}}{1+e^{-\beta\mathbf{x}}}$, with β setting to 1.

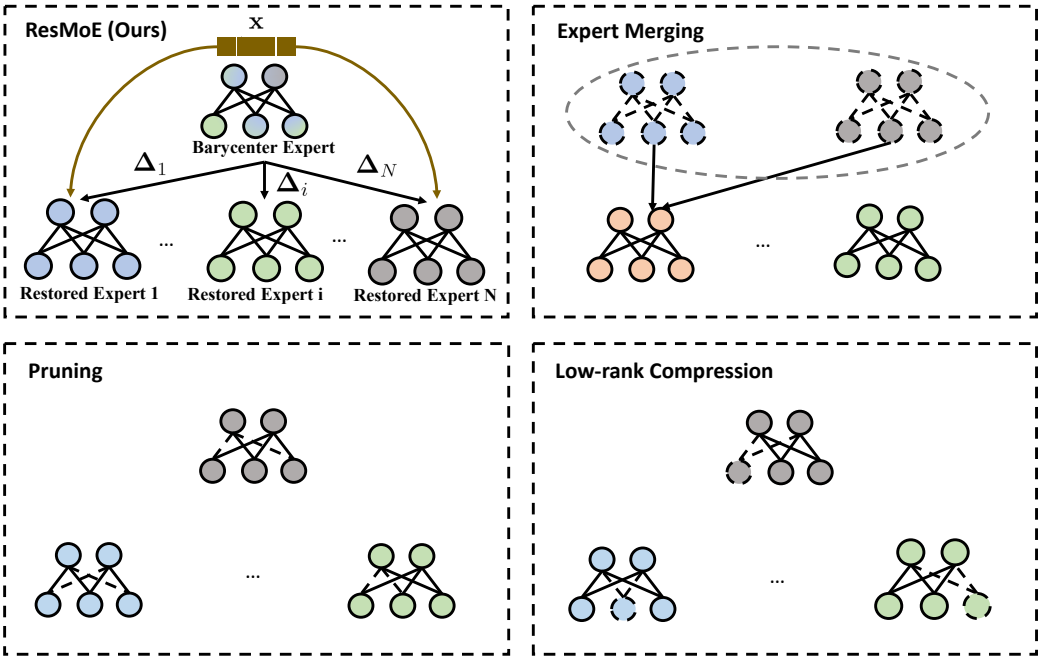

Figure 4. Comparisons between ResMoE and baselines. Dash lines denote the connections or neurons are pruned. Yellow lines denote the flow of data. In our proposed ResMoE, the input $\mathbf{x}$ is fed into selected restored experts. For baselines, the input is directed to their compressed experts, resulting in a similar flow. We have omitted the flow of baselines for convenience. Detailed baseline comparisons are shown in Sec. C.1

Similarly, for Mixtral and DeepSeekMoE, the extraction objective is:

$$\min_{\substack{\mathbf{W}_\omega^{(1)},\mathbf{b}_\omega^{(1)},\mathbf{W}_\omega^{(3)},\mathbf{b}_\omega^{(3)},\mathbf{W}_\omega^{(2)} \\ \mathbf{T}_k\in\mathcal{P},k\in[N]}} \frac{1}{N}\sum_{k=1}^{N}\left[\left\|\mathbf{T}_k\left(\mathbf{W}_k^{(1)},\mathbf{b}_k^{(1)},\mathbf{W}_k^{(3)},\mathbf{b}_k^{(3)}\right)-\left(\mathbf{W}_\omega^{(1)},\mathbf{b}_\omega^{(1)},\mathbf{W}_\omega^{(3)},\mathbf{b}_\omega^{(3)}\right)\right\|_F^2\right.$$
$$\left.+\left\|\mathbf{W}_k^{(2)}\mathbf{T}_k^{\mathrm{T}}-\mathbf{W}_\omega^{(2)}\right\|_F^2\right].$$

Then we can extend the $\mathbf{W}_k$ to:

$$\mathbf{W}_k=\left[\mathbf{W}_k^{(1)},\mathbf{b}_k^{(1)},\mathbf{W}_k^{(3)},\mathbf{b}_k^{(3)},(\mathbf{W}_k^{(2)})^{\mathrm{T}}\right],$$

and ResMoE can then be similary applied to Mixtral and DeepSeekMoE.