RECONSTRUCTION-OPTIMIZED EXPERT PRUNING FOR MIXTURE-OF-EXPERTS LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Mixture-of-Experts (MoE) models reduce computation through sparse expert activation, but deployment still requires storing a large number of expert parameters. We study post-training expert pruning using task-domain calibration data as a subset-selection problem: given a pretrained MoE layer, select a fixed number of experts that best reconstruct its original output. Exact subset search is combinatorial and becomes intractable for modern MoE models with many experts per layer. We develop OptEP, a scalable continuous relaxation that directly optimizes differentiable expert-selection variables, followed by a lightweight reconstruction-based update of the retained expert weights. Across multiple MoE architectures and tasks spanning mathematical reasoning and code generation, our approach improves over the strongest existing expert-pruning baselines, with absolute accuracy gains of up to 15.05% on Qwen3-30B-A3B, 14.55% on Qwen3-235B-A22B, 11.06% on GLM-4.5-Air, and 16.20% on Mixtral-8x7B. The improvements are particularly pronounced under aggressive pruning, demonstrating that reconstruction-driven expert selection and weight adaptation can preserve substantially more of the original model's performance while reducing the number of retained experts.

1 INTRODUCTION

Mixture-of-Experts (MoE) models increase model capacity without proportionally increasing per-token computation by routing each token to only a small subset of expert networks (Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2022; Du et al., 2022). This architecture underlies open-weight language models such as Mixtral, DeepSeek-MoE, Qwen, and OLMoE (Jiang et al., 2024; Dai et al., 2024; Yang et al., 2025; Muennighoff et al., 2025). However, the computational savings from sparse activation do not eliminate the memory burden of storing the complete expert pool, even though each token uses only a fraction of it. As modern MoEs scale from tens of billions to trillions of parameters, this growing parameter footprint poses a substantial deployment challenge, motivating methods that reduce the number of experts while preserving model capabilities.

Expert pruning addresses this challenge by permanently removing a subset of experts after training. The central challenge is to identify a smaller collection that preserves the capabilities of the original model. Common strategies assign each expert an importance score based on activation frequency, router probability, output magnitude, or related local statistics, and retain the highest-scoring experts (Muzio et al., 2024; Dong et al., 2026; Lasby et al., 2026). These approaches are simple and scalable, but selecting experts individually according to their scores does not directly optimize the quality of the retained subset as a whole.

The usefulness of an expert depends on the contributions of the other experts retained alongside it. Experts with high individual scores may provide overlapping contributions, while a lower-scoring expert may supply complementary information that is difficult for the others to reproduce. Consequently, a collection of individually high-scoring experts need not be the collection that best preserves the original layer's computation. This motivates a joint selection criterion: evaluate how well the retained experts collectively reconstruct the output of the full MoE layer on representative calibration data.

Reconstruction-based selection makes this objective explicit, but introduces a combinatorial optimization problem. Selecting s experts from a pool of E admits $\binom{E}{s}$ possible subsets. Prior work evaluates candidate subsets exhaustively (over the $\binom{E}{s}$ pool) using layer-output reconstruction error (Lu et al., 2024), an approach that is practical for small expert pools but becomes prohibitive as the number of

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

experts per layer and/or the number of model layers grows. The challenge is therefore to make joint, reconstruction-driven selection scalable without exhaustively enumerating expert combinations.

Expert selection also leaves a second opportunity for improvement. After pruning, the retained experts operate within a different expert pool from the one in which they were pretrained, and their original parameters are not optimized to compensate for the functions of removed experts. Adapting the retained experts after selection can therefore recover performance that would otherwise be lost. This motivates treating expert selection and post-pruning adaptation as two connected optimization problems, both guided by reconstruction of the original layer output.

We propose **OptEP**, a two-stage framework that follows this principle. First, we formulate expert pruning as layerwise subset selection and introduce a differentiable reconstruction surrogate that jointly optimizes expert-selection variables. This allows selection to account for the combined contribution of the retained experts while avoiding exhaustive enumeration of expert subsets. Our flexible continuous optimization framework enables us to explore different forms of the reconstruction loss, that better capture the effect of infrequently activated experts through weighing different tokens in the calibration set. Second, after fixing the retained subset of experts, we optimize its expert weights using a layerwise reconstruction objective (with additional regularization). Together, these stages address both which experts to retain and how to adapt them after pruning, without requiring prohibitively expensive full-model retraining.

Our main contributions are:

- We formulate expert pruning as joint reconstruction-based subset selection and develop a scalable differentiable optimization approach that directly optimizes layer-output reconstruction without combinatorial enumeration. In addition, we study a token-weighted reconstruction loss that captures the effect of tokens routed through infrequently activated experts.
- We introduce a reconstruction-based expert-weight update that adapts retained experts after pruning with the goal of recovering the accuracy lost from expert selection.
- We evaluate our framework across multiple MoE language models on mathematical reasoning and code-generation tasks. At the most aggressive pruning setting evaluated for each model–task pair, our method with expert-weight updates achieves absolute accuracy improvements of up to 15.05% on Qwen3-30B-A3B, 14.55% on Qwen3-235B-A22B, 11.06% on GLM-4.5-Air, and 16.20% on Mixtral-8x7B over the strongest competing baseline at the same retention budget. The pruned models remain highly performant. For example, Qwen3-235B-A22B retains 97% of its unpruned performance on a math benchmark after pruning 75% of its experts and applying the weight update.

2 Related Work

Importance-based expert pruning SEER-MoE (Muzio et al., 2024) prunes experts by activation frequency. EASY-EP (Dong et al., 2026) scores experts on domain-specific demonstrations by combining router weights and expert-output magnitudes with token-importance weights derived from input–output representation similarity. Motivated by a reconstruction-error bound, REAP (Lasby et al., 2026) averages router-weighted expert output magnitudes over tokens activating each expert, reducing frequency bias against experts with strong but infrequent contributions. Shapley-based scoring (Huang et al., 2026) accounts for interactions through marginal contributions across sampled subsets. These methods rank individual experts rather than directly optimizing the reconstruction error of the retained set.

Reconstruction- and search-based expert selection NAEE (Lu et al., 2024) exhaustively searches expert subsets to minimize the discrepancy between original and pruned MoE-layer outputs on calibration data. This is tractable for Mixtral-8x7B's eight experts per layer, but scales combinatorially to larger expert pools. EEP (Liu et al., 2024) uses evolutionary search for pruning configurations and subsequent merging weights. Our continuous relaxation optimizes a smooth reconstruction surrogate with pretrained router and expert weights fixed, then discretizes the selection to a fixed expert budget per layer.

Layer-adaptive pruning Prior work also considers non-uniform pruning across layers, using expert- and layer-level importance (Yang et al., 2024; Bai et al., 2026; Yoon & Choi, 2026) or search over layer-wise pruning budgets (Liu et al., 2026). In contrast, we apply uniform pruning, retaining the same number of experts in every layer.

Expert diversity and merging Clustering-based pruning preserves complementary experts by grouping similar experts (Zhang et al., 2025) or comparing behavior across task domains (Hu et al., 2026). Our merging baseline, M-SMoE (Li et al., 2024), aligns expert neurons, groups experts around frequently activated representatives using routing similarity, and averages their parameters with activation-frequency weights. This consolidates knowledge into fewer experts rather than simply discarding it. DERN (Zhou et al., 2025) transfers knowledge through neuron-level recombination. We instead retain an explicit expert subset and adapt its weights by reconstruction.

Post-pruning adaptation Prior recovery methods include regularization-based fine-tuning in SEER-MoE (Muzio et al., 2024), corrective fine-tuning during iterative pruning (Jaiswal et al., 2025), and Fisher-guided weight compensation in COMPEL (Yoon & Choi, 2026). Our separate recovery stage reconstructs original layer outputs while regularizing retained experts toward their pretrained weights. This connects to layerwise weight-pruning methods such as Optimal Brain Compression, SparseGPT, and ALPS (Frantar & Alistarh, 2022; 2023; Meng et al., 2024a), while expert removal additionally changes both the available functions and their routing weights. Broader compression strategies also prune within experts or remove layers and blocks (He et al., 2024).

3 METHOD

Background Let $\mathcal{E} = \{1, \dots, E\}$ denote the experts in one MoE layer, $K \geq 1$ of which get activated for each token. Let $\boldsymbol{\theta}_e$ denote the collection of weight matrices parameterizing expert e, and let $f_e(\boldsymbol{x}; \boldsymbol{\theta}_e)$ be the output of expert e for layer input activation $\boldsymbol{x}$ corresponding to a given token. Let $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_e\}_{e \in \mathcal{E}}$ denote the collection of all experts weights for the current layer. We omit the layer index for notational simplicity; all quantities and optimization problems in this section are defined for a given layer. For each token $\boldsymbol{x}$, the pretrained router produces routing scores $\boldsymbol{a}(\boldsymbol{x}) \in \mathbb{R}^E_{\geq 0}$. Let $\mathcal{T}_K(\boldsymbol{a}(\boldsymbol{x})) \subseteq \mathcal{E}$ contain the K experts with the largest router scores for input $\boldsymbol{x}$ for a given token. For all models considered in this work, the routing scores are normalized over the selected top-K experts. GLM-4.5-Air additionally contains one always-active shared expert. We prune only the routed experts and keep the shared expert unchanged; for notational simplicity, $\mathcal{E}$ refers only to the set of routed experts throughout this section. Formally, the output of the original layer is given as

$$\boldsymbol{y}(\boldsymbol{x}) = \sum_{e \in \mathcal{T}_K(\boldsymbol{a}(\boldsymbol{x}))} p_e^{(K)}(\boldsymbol{x}) f_e(\boldsymbol{x}; \boldsymbol{\theta}_e), \quad p_e^{(K)}(\boldsymbol{x}) = \frac{a_e(\boldsymbol{x})}{\sum_{j \in \mathcal{T}_K(\boldsymbol{a}(\boldsymbol{x}))} a_j(\boldsymbol{x})}. \tag{1}$$

We assume a calibration set contains M sequences, each consisting of a prompt and completion. Let T_m denote the length of calibration sequence m, for $1 \leq m \leq M$, and let $\{\boldsymbol{x}_n^{(m)}\}_{n=1}^{T_m}$ denote the corresponding input activations to the layer for the T_m calibration tokens used in the optimization.

Expert subset selection We follow a layerwise pruning approach (Frantar & Alistarh, 2022), pruning each MoE layer independently. Layerwise methods are widely used in LLM pruning (Frantar & Alistarh, 2023; Sun et al., 2024; Meng et al., 2024a), offering high accuracy at substantially lower cost than end-to-end pruning. For a given layer, let $z_e \in \{0, 1\}$ indicate whether expert $e \in \mathcal{E}$ is retained, where $z_e = 1$ denotes retention and $z_e = 0$ denotes expert removal. Let $\widehat{\boldsymbol{y}}^{(K)}(\boldsymbol{x}; \boldsymbol{z}, \boldsymbol{\Theta})$ denote the resulting layer output for token $\boldsymbol{x}$. Pruning according to $\boldsymbol{z}$ is equivalent to replacing the router scores $\boldsymbol{a}(\boldsymbol{x})$ with $\boldsymbol{z} \odot \boldsymbol{a}(\boldsymbol{x})$, where $\odot$ denotes element-wise multiplication. Thus, the pruned MoE layer output is given by

$$\widehat{\boldsymbol{y}}^{(K)}(\boldsymbol{x}; \boldsymbol{z}, \boldsymbol{\Theta}) = \sum_{e \in \mathcal{T}_K(\boldsymbol{a}(\boldsymbol{x}) \odot \boldsymbol{z})} \pi_e(\boldsymbol{x}; \boldsymbol{z}) f_e(\boldsymbol{x}; \boldsymbol{\theta}_e), \quad \pi_e(\boldsymbol{x}; \boldsymbol{z}) = \frac{z_e a_e(\boldsymbol{x})}{\sum_{j \in \mathcal{T}_K(\boldsymbol{a}(\boldsymbol{x}) \odot \boldsymbol{z})} z_j a_j(\boldsymbol{x})}. \tag{2}$$

Given a pruning budget s, with $K \leq s < E$, we retain exactly s experts per layer by minimizing the reconstruction loss between the full and pruned models on the calibration set. We formulate the layerwise objective as the subset-selection problem:

$$\min_{\boldsymbol{z}} \ \frac{1}{2M} \sum_{m=1}^{M} \frac{1}{T_m} \sum_{n=1}^{T_m} \mathcal{L}\left(\boldsymbol{y}(\boldsymbol{x}_n^{(m)}), \widehat{\boldsymbol{y}}^{(K)}(\boldsymbol{x}_n^{(m)}; \boldsymbol{z}, \boldsymbol{\Theta})\right) \ \text{ s.t. } \ z_e \in \{0, 1\} \ \forall e \in \mathcal{E}, \ \sum_{e \in \mathcal{E}} z_e = s \tag{3}$$

where $\mathcal{L}(\boldsymbol{u}, \boldsymbol{v})$ quantifies the distance between two vectors. In particular, in this paper we assume $\mathcal{L}(\boldsymbol{u}, \boldsymbol{v}) = \|\boldsymbol{u} - \boldsymbol{v}\|_2^\alpha$, where $\alpha \in \{1, 2\}$ is a hyperparameter. The $1/T_m$ normalization gives each

calibration sequence equal total weight, preventing longer sequences from dominating the reconstruction objective. Thus, (3) selects the s experts that minimize aggregate layerwise reconstruction error over the calibration set, with $\boldsymbol{\Theta}$ fixed.

Continuous optimization of expert selection: OptEP (uw) Directly optimizing (3) requires a combinatorial search over expert subsets which is expensive for large expert pools. To obtain a differentiable optimization formulation, we relax the binary selection variables to $z \in [0,1]^E$ and replace the discontinuous Top-K pruned-layer output in (2) with a smooth surrogate

$$\widetilde{\boldsymbol{y}}(\boldsymbol{x};\boldsymbol{z},\boldsymbol{\Theta}) = \sum_{e\in\mathcal{E}} \tilde{\pi}_e(\boldsymbol{x};\boldsymbol{z}) f_e(\boldsymbol{x};\boldsymbol{\theta}_e), \quad \text{where } \tilde{\pi}_e(\boldsymbol{x};\boldsymbol{z}) = \frac{z_e a_e(\boldsymbol{x})}{\sum_{j\in\mathcal{E}} z_j a_j(\boldsymbol{x})}. \tag{4}$$

This leads to the following continuous optimization problem:

$$\min_{\boldsymbol{z}} \ \frac{1}{2M}\sum_{m=1}^{M}\frac{1}{T_m}\sum_{n=1}^{T_m}\mathcal{L}\left(\boldsymbol{y}(\boldsymbol{x}_n^{(m)}), \widetilde{\boldsymbol{y}}(\boldsymbol{x}_n^{(m)};\boldsymbol{z},\boldsymbol{\Theta})\right) \quad \text{s.t.} \ \ z_e \in [0,1]\ \forall e \in \mathcal{E}. \tag{5}$$

We optimize (5) while keeping the pretrained router and expert weights $\boldsymbol{\Theta}$ fixed. We do not enforce the cardinality constraint $\sum_e z_e = s$ during continuous optimization; instead, after optimization, we discretize the solution by retaining the s experts with the largest values of z_e. After discretization, the resulting model remains a standard K-sparsely activated MoE, with s rather than E experts per layer.

Weighted reconstruction objective: OptEP (w) In addition to the formulation in (5), we consider a weighted variant, OptEP (w), designed to reduce the dominance of frequently activated experts in the reconstruction objective. Tokens routed through infrequently used experts receive larger weights, giving these experts greater influence during selection and helping preserve specialized or complementary behavior that may otherwise be underrepresented in the calibration data. Specifically, let n_e denote the number of calibration tokens routed to expert e by the unpruned model in the current layer. For a token with layer input $\boldsymbol{x}$, we define its weight as

$$w(\boldsymbol{x}) = \sum\nolimits_{e\in\mathcal{T}_K(\boldsymbol{a}(\boldsymbol{x}))} 1/n_e, \tag{6}$$

where we sum over the experts active for that token. The weighted reconstruction objective based optimization problem is:

$$\min_{\boldsymbol{z}} \ \frac{1}{2M}\sum_{m=1}^{M}\frac{1}{T_m}\sum_{n=1}^{T_m} w(\boldsymbol{x}_n^{(m)})\cdot\mathcal{L}\left(\boldsymbol{y}(\boldsymbol{x}_n^{(m)}), \widetilde{\boldsymbol{y}}(\boldsymbol{x}_n^{(m)};\boldsymbol{z},\boldsymbol{\Theta})\right) \ \text{ s.t. } \ z_e \in [0,1]\ \forall e \in \mathcal{E}. \tag{7}$$

Appendix A provides a theoretical interpretation of this weighting. Up to a constant normalization, it converts the token-level reconstruction objective into an expert-balanced objective that assigns equal relative weight to each expert-conditioned reconstruction loss. This removes the direct dependence of the aggregation weights on expert activation frequency and yields simple frequency-independent bounds under reweightings of the expert-conditioned losses.

Reconstruction-based expert weight update: EWU To recover accuracy lost during pruning, we update the weights of the retained experts to compensate for those removed. Let $\boldsymbol{\Theta}' = \{\boldsymbol{\theta}'_e : e \in \mathcal{E}, z_e = 1\}$ denote the updated weights of the retained experts. With the selected subset $\boldsymbol{z}$ fixed, we optimize only the weights $\boldsymbol{\theta}'_e$ of selected experts ($z_e = 1$) using the layerwise objective:

$$\min_{\boldsymbol{\Theta}'} \ \frac{1}{2M}\sum_{m=1}^{M}\frac{1}{T_m}\sum_{n=1}^{T_m}\mathcal{L}\left(\boldsymbol{y}(\boldsymbol{x}_n^{(m)}), \widehat{\boldsymbol{y}}^{(K)}(\boldsymbol{x}_n^{(m)};\boldsymbol{z},\boldsymbol{\Theta}')\right) + \lambda \sum_{\substack{e\in\mathcal{E}\\ z_e=1}} \|\boldsymbol{\theta}'_e - \boldsymbol{\theta}_e\|_F^2, \tag{8}$$

where $\lambda \geq 0$ is a pre-specified regularization parameter to avoid overfitting (Frantar & Alistarh, 2023; Meng et al., 2024b). For a fixed layer, the router, retained expert subset, layer inputs are held fixed, and only the retained expert weights $\boldsymbol{\Theta}'$ are optimized. The objective in (8) is differentiable in $\boldsymbol{\Theta}'$.

Layer Inputs for Expert Selection and EWU For the expert-selection objectives in (5) and (7), $\boldsymbol{x}_n^{(m)}$ denotes the layer input obtained from a forward pass of the full, unpruned model. Expert selection is therefore performed independently across layers: pruning decisions made in one layer do not affect the inputs or expert selection in subsequent layers. We use this independent layerwise

setup to allow comparisons with prior expert-pruning baselines, which likewise perform pruning independently across layers (Dong et al., 2026; Lasby et al., 2026).

In contrast, EWU is performed sequentially across layers. When optimizing a given layer, $\boldsymbol{x}_n^{(m)}$ is obtained by forwarding the calibration data through the pruned model, including the updated weights of all preceding layers. Consequently, weight updates to earlier layers affect the inputs used to optimize subsequent layers.

Joint versus fixed-score expert selection Importance-based pruning methods assign each expert a fixed score and retain the highest-scoring experts. In Appendix B, we show that such a fixed-score top-s rule arises naturally from a local first-order approximation of the OptEP reconstruction objective: the resulting expert score is the negative gradient of the reconstruction loss with respect to its selection variable at a fixed reference point. OptEP instead optimizes the full nonlinear continuous reconstruction surrogate, so these local scores evolve as the selection variables and reconstructed mixture change. Hence the marginal value of an expert depends on the experts currently emphasized, reducing the value of redundant experts while preserving the value of complementary ones.

4 Experiments

Experimental goals We evaluate whether reconstruction-optimized expert selection improves downstream performance over existing expert-pruning methods across model scales, task domains, and pruning levels. We further study (i) whether inverse-frequency token weighting improves expert selection, (ii) whether reconstruction-based expert-weight updates recover additional performance after pruning, and (iii) whether our continuous relaxation can match the performance of exhaustive reconstruction-based subset selection when exact enumeration is feasible.

4.1 Experimental Setup

Models We evaluate Qwen3-30B-A3B, Qwen3-235B-A22B (Yang et al., 2025), GLM-4.5-Air (Zeng et al., 2025), and Mixtral-8x7B-v0.1 (Jiang et al., 2024). The two Qwen models contain 128 routed experts in each of their 48 and 94 MoE layers, respectively, with Top-8 routing. GLM-4.5-Air contains 46 transformer layers, of which the first is dense and the remaining 45 are MoE layers, with 128 routed experts and one always-active shared expert per MoE layer. It uses Top-8 routing; we prune only the routed experts and leave the shared expert unchanged. Mixtral-8x7B-v0.1 contains 8 experts in each of its 32 MoE layers with Top-2 routing.

For each task domain (math or coding), we prune the same proportion of routed experts from every MoE layer, producing a task-specific model with the same number of layers as the original model but fewer routed experts per layer. For Qwen, we retain 32, 48, and 64 experts per layer (25%, 37.5%, and 50%) for math, and 48, 64, and 96 experts (37.5%, 50%, and 75%) for coding. For GLM-4.5-Air, we retain 32, 48, and 64 routed experts per layer for math, and 64, 80, and 96 routed experts per layer for coding. For Mixtral-8x7B, we retain 4 or 6 of the 8 experts per layer (50% or 75%). We do not consider more aggressive pruning settings because, for each model–task pair, retaining fewer experts than the lowest setting reported above leads to substantial performance degradation.

Calibration We use domain-specific calibration for expert selection: models evaluated on math benchmarks are pruned using traces generated from math problems, while those evaluated on coding benchmarks use traces from code-generation problems (Lucas et al., 2026). The same base model is used to generate the calibration traces; this model is then pruned, and the resulting pruned model is used for final evaluation. For Qwen and GLM, we generate full-model calibration traces from OpenR1-Math-220k (Hugging Face, 2025) for math and Codeforces (Penedo et al., 2025) for coding. Each calibration sequence consists of the prompt concatenated with the model-generated completion, including its reasoning trajectory. Because Mixtral-8x7B performs poorly on MATH-500 and LiveCodeBench, these benchmarks provide limited resolution for comparing pruning methods. We therefore use GSM8K (Cobbe et al., 2021) training examples for math and MBPP (Austin et al., 2021) for coding, where differences between pruning methods are more readily measurable.

For all models and domains, we retain only untruncated samples for which the full model produces the correct answer. This ensures that expert selection is based on successful model trajectories rather than activations arising from failed reasoning or incorrect generations. Expert selection uses a fixed calibration-token budget; the sampled examples vary across seeds while the total number of calibration tokens remains fixed. The same calibration sequences used for expert selection are

reused for the expert weight update, so both stages operate on an identical calibration distribution. Additional details on the calibration sets, including the token budget and number of sequences used, are provided in Appendix C.1.

Evaluation For Qwen and GLM, the pruned models selected using math calibration traces are evaluated zero-shot on MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024), while those selected using coding calibration traces are evaluated on LiveCodeBench (LCB v4–v5) (Jain et al., 2025). We report mean pass@1 across seeds. Variation across seeds reflects both calibration-set sampling and stochastic evaluation sampling, with evaluation temperature fixed at 0.6. For Mixtral-8x7B, models pruned using GSM8K math calibration traces are evaluated on the GSM8K test set, while models pruned using MBPP coding calibration traces are evaluated on held-out MBPP examples. We report mean pass@1 across seeds. Because Mixtral-8x7B uses greedy decoding (temperature 0), there is no inference-time sampling variability. For GSM8K, variation across seeds therefore reflects only calibration-set selection; for MBPP, it additionally reflects variation in the seed-specific held-out evaluation split. Additional evaluation details, including evaluation set sizes and decoding settings, are provided in Appendix C.2.

Baselines We compare against random pruning, routing-frequency pruning (Muzio et al., 2024), EASY-EP (Dong et al., 2026), REAP (Lasby et al., 2026) and NAEE (Lu et al., 2024). These baselines, as well as our method, use uniform pruning, retaining the same number of experts in every MoE layer. NAEE exhaustively enumerates candidate expert subsets, making it practical for Mixtral-8x7B, which has only 8 experts per layer, but computationally infeasible for the Qwen models with 128 experts per layer. We therefore evaluate NAEE only on Mixtral-8x7B. We additionally compare against M-SMoE (Li et al., 2024), which combines expert pruning with expert merging. We also include DiEP (Bai et al., 2026), a non-uniform pruning method that allows the number of retained experts to vary across MoE layers. We were unable to successfully run the available DiEP implementation on the Qwen and GLM models due to implementation and scalability limitations, so we report DiEP results only for Mixtral-8x7B. Additional details on the baselines are provided in Appendix C.3.

For a given model, task, and retention level, all methods use the same calibration data and token budget and are evaluated under the same downstream protocol, controlling for differences in data, pruning budget, and evaluation setup.

Implementation and Compute Details We compare the base OptEP (uw) objective, its inverse-frequency-weighted variant OptEP (w), and their corresponding expert-weight-update variants against the baselines described above. The optimization details of these pruning and weight update methods are in Appendix C.4. Qwen3-235B-A22B and GLM-4.5-Air experiments used 8 H100 GPUs per seed, while Qwen3-30B-A3B and Mixtral-8x7B experiments used 4 H100 GPUs per seed.

4.2 Main Results

Results Tables 1, 2, 3, and 4 compare the mean performance of OptEP (uw), OptEP (w), OptEP (w) + EWU, and the baseline methods. OptEP (w) + EWU applies the expert-weight update to the experts selected by OptEP (w). All reported values are averaged across experimental seeds. In each table, bold values indicate the best-performing method and underlined values indicate the second-best. Full results, including standard errors, are provided in Appendix D.

Pure Pruning Performance We treat OptEP (w) as our primary pruning method, as it performs better overall than the unweighted OptEP (uw) variant. Among methods that perform pure expert pruning without modifying expert weights, OptEP (w) achieves the best performance in 18 of 22 settings. Its advantages are largest under aggressive pruning, with improvements over the strongest prior pruning baseline of up to 10.45% on Qwen3-30B-A3B, 10.32% on Qwen3-235B-A22B, 4.60% on GLM-4.5-Air, and 1.61% on Mixtral-8x7B. All the reported differences are absolute percentage points. On Mixtral-8x7B, OptEP (w) performs best in all four settings, although the gains are smaller, potentially because the model has only 8 experts per layer and therefore offers less room for improved subset selection. In the three settings where OptEP (w) is outperformed by a prior baseline, the differences are at most 0.5%.

More broadly, at least one of the two reconstruction-based variants OptEP (uw) or OptEP (w) outperforms the strongest prior pruning baseline in 21 of 22 settings. Overall, these results show that reconstruction-based expert selection performs strongly across models and tasks, with the largest gains under aggressive pruning.

At less aggressive pruning levels, the reconstruction-based methods closely approach the performance of the unpruned model. In some cases they even exceed it on average—e.g, on Qwen3-30B-A3B coding at $s = 96$, OptEP (w) achieves 61.44% compared with 60.57% for the full model.

Expert Weight Update Gains Applying expert weight updates after pruning improves performance in 19 of 22 settings compared with pruning alone. On Qwen3-30B-A3B, OptEP (w) + EWU outperforms OptEP (w) in 5 of 6 settings, with gains of up to 4.60% absolute improvement. On Qwen3-235B-A22B, OptEP (w) + EWU improves over OptEP (w) in all 6 settings, with gains of up to 4.23%. On GLM-4.5-Air, OptEP (w) + EWU improves over OptEP (w) in 4 out of 6 settings, with gains up to 6.46%. On Mixtral-8x7B, OptEP (w) + EWU improves over OptEP (w) in all 4 settings, with gains of up to 15.27%.

Across all four models, OptEP (w) + EWU achieves the best performance in 20 of 22 settings relative to the existing baselines. The absolute improvements reach up to 11.33% on Qwen3-30B-A3B math, 15.05% on Qwen3-30B-A3B coding, 10.60% on Qwen3-235B-A22B math, 14.55% on Qwen3-235B-A22B coding, 11.06% on GLM-4.5-Air math, 6.34% on GLM-4.5-Air coding, 16.20% on Mixtral-8x7B math, and 10.80% on Mixtral-8x7B coding. The benefits of EWU are generally larger under more aggressive pruning, where removing experts induces a greater mismatch between the retained experts and the original layer output. This suggests that the reconstruction-based update helps the retained experts compensate for capacity lost during pruning, making expert selection and post-pruning adaptation complementary.

Table 1: Expert pruning results for Qwen3-30B-A3B on math (MATH-500 pass@1) and coding (LiveCodeBench pass@1). Results are mean accuracy in % over three calibration seeds.

Method	Math (MATH-500)			Coding (LiveCodeBench)		
	s=32	s=48	s=64	s=48	s=64	s=96
Random	0.00	0.00	0.00	0.00	0.00	0.00
Frequency	1.40	19.27	66.33	0.00	1.87	**61.44**
EASY-EP	31.73	93.73	94.60	17.54	54.73	61.07
REAP	76.07	91.93	94.73	38.43	52.49	59.83
M-SMoE	1.33	1.00	1.07	0.00	0.00	59.83
OptEP (uw)	**88.40**	93.40	94.67	50.50	56.22	60.57
OptEP (w)	83.47	93.53	95.47	48.88	54.60	**61.44**
OptEP (w) + EWU	87.40	**94.40**	**96.13**	**53.48**	**56.59**	61.19
Full model (128 experts)		96.40			60.57	

Table 2: Expert pruning results for Qwen3-235B-A22B on math (MATH-500 pass@1) and coding (LiveCodeBench pass@1). Results are mean accuracy in % over three calibration seeds.

Method	Math (MATH-500)			Coding (LiveCodeBench)		
	s=32	s=48	s=64	s=48	s=64	s=96
Random	0.67	1.13	0.60	0.00	0.00	0.00
Frequency	0.07	0.27	93.20	0.00	45.52	64.43
EASY-EP	71.33	93.20	95.20	41.67	56.97	61.19
REAP	84.07	93.60	96.07	41.17	61.69	64.68
M-SMoE	0.67	0.13	60.07	0.00	27.11	64.68
OptEP (uw)	87.80	92.53	95.53	43.04	62.31	65.17
OptEP (w)	91.87	95.27	96.13	51.99	62.44	64.18
OptEP (w) + EWU	**94.67**	**96.13**	**96.47**	**56.22**	**62.94**	**66.42**
Full model (128 experts)		97.60			66.67	

Table 3: Expert pruning results for GLM-4.5-Air on math (MATH-500 pass1) and coding (LiveCodeBench pass1). Results are mean accuracy in % over three calibration seeds.

Method	Math (MATH-500)			Coding (LiveCodeBench)		
	s=32	s=48	s=64	s=64	s=80	s=96
Random	1.40	5.13	27.07	0.50	3.48	22.39
Frequency	26.47	56.93	74.87	3.23	14.68	32.21
EASY-EP	71.80	92.07	95.53	45.77	54.85	**64.30**
REAP	75.87	<u>93.73</u>	96.60	49.88	59.33	<u>64.18</u>
M-SMoE	2.47	27.53	67.27	1.24	9.33	27.49
OptEP (uw)	77.73	92.53	**97.07**	51.99	<u>62.56</u>	62.06
OptEP (w)	<u>80.47</u>	<u>93.73</u>	96.20	<u>52.61</u>	**62.69**	**64.30**
OptEP (w) + EWU	**86.93**	**94.93**	<u>97.00</u>	**56.22**	61.82	62.06
Full model (128 experts)		97.87			65.30	

Table 4: Expert pruning results for Mixtral-8x7B-v0.1 on math (GSM8K flexible extraction) and coding (MBPP standard pass@1). Results are mean accuracy in % over five seeds.

Method	Math (GSM8K)		Coding (MBPP)	
	s=4	s=6	s=4	s=6
Random	3.08	20.15	7.74	40.25
Frequency	8.75	8.96	18.71	37.51
EASY-EP	7.01	25.26	28.70	47.57
REAP	11.45	23.55	25.07	44.49
M-SMoE	10.63	<u>30.51</u>	24.45	38.05
NAEE brute-force	11.96	24.96	28.53	43.87
DiEP	10.19	24.90	16.54	41.91
OptEP (uw)	<u>14.04</u>	24.58	27.99	47.48
OptEP (w)	12.89	25.32	<u>30.31</u>	<u>47.61</u>
OptEP (w) + EWU	**28.16**	**32.90**	**39.50**	**50.19**
Full model (8 experts)	47.61		59.34	

4.3 Ablation Studies

Effect of Token Weighting OptEP (w) improves over OptEP (uw) in 3 of 6 settings on Qwen3-30B-A3B, with gains of up to 0.87%, and in 5 of 6 settings on Qwen3-235B-A22B, with substantially larger gains of up to 8.95%. On GLM-4.5-Air, OptEP (w) improves over OptEP (uw) in 5 out of 6 settings, with gains up to 2.74%. On Mixtral-8x7B, OptEP (w) improves over OptEP (uw) in 3 of 4 settings, with gains of up to 2.32%. The largest drop from the unweighted to the weighted variant occurs on Qwen3-30B-A3B for both math and coding at the lowest sparsity levels; however, both variants still outperform existing baselines by substantial margins in these settings.

One possible explanation for the larger gains on Qwen3-235B-A22B is that, under the same fixed calibration-token budget, its longer generations result in fewer calibration sequences than for Qwen3-30B-A3B, as detailed in Appendix C.1. The longer calibration sequences may cause frequently activated experts to dominate the unweighted reconstruction objective. The weighting partially counteracts this imbalance by assigning greater weight to tokens that activate less frequently used experts, potentially helping preserve experts that are important despite appearing less often in the calibration data. This suggests that the weighting may be particularly beneficial for models that produce long, verbose reasoning traces.

Alternative Token Weighting In Appendix D, we report the full results for OptEP (w-alt), which uses a slightly different token-weighting scheme but otherwise follows the same approach as OptEP (w). Across the four models, the maximum absolute difference between OptEP (w) and OptEP (w-alt) is 3.20% on GLM-4.5-Air, with differences below 2.6% on the other four models. Overall, the

two weighting schemes yield similar performance, suggesting that the benefits of inverse-frequency weighting are not highly sensitive to the precise weighting formulation.

Comparison with Brute-Force Subset Selection On Mixtral-8x7B, OptEP (w) achieves accuracy comparable to or higher than NAEE, with gains of up to 3.74%. NAEE (Lu et al., 2024) performs exhaustive search over expert subsets using a reconstruction objective similar to ours, but with two differences: it does not normalize the loss per sequence, and it computes the Frobenius-norm reconstruction error over batches of calibration sequences, whereas our objective computes the ℓ_2 reconstruction error over individual calibration tokens.

To provide a more direct comparison, Table 11 in Appendix D additionally reports two brute-force variants that optimize the exact discrete counterparts of our objectives. OptEP (uw) brute force exactly solves the subset-selection problem in (3), while OptEP (w) brute force exactly solves the corresponding weighted objective. Across the evaluated settings, OptEP (uw) and OptEP (w) perform similarly to their respective brute-force counterparts. Across the evaluated Mixtral settings, OptEP (uw) differs from its exact brute-force counterpart by at most 2.96%, while OptEP (w) differs from its corresponding brute-force solution by at most 2.53% absolute difference. These results suggest that the proposed continuous surrogate provides a useful approximation to the underlying discrete subset-selection objectives while avoiding the combinatorial cost of exhaustive search.

Expert Weight Updates Across Pruning Variants To assess whether EWU depends on the expert-selection objective, we apply it after both OptEP (uw) and OptEP (w) across all evaluated models in Appendix D. EWU improves downstream performance in 20 of 22 settings when applied after OptEP (uw) and in 19 of 22 settings when applied after OptEP (w), indicating that its benefits are not specific to the weighted selection objective. In most cases, the pruning method with stronger pre-update performance also performs better after EWU, although this relationship does not hold universally. When the pre-update performances are close, EWU can change their relative ordering, whereas larger gaps after pruning are generally preserved after the update. More broadly, EWU can be applied after any expert-selection baseline, although substantial performance gaps introduced during pruning are unlikely to be reversed by the subsequent weight update.

5 Limitations

Our experiments focus on four open-weight MoE model families and on mathematical reasoning and code-generation tasks, so the extent to which the results generalize to other architectures and domains remains to be evaluated. We additionally use domain-specific calibration data consisting of correct trajectories generated by the full model; performance under noisier, unlabeled, or out-of-domain calibration data remains to be studied. We use uniform pruning, retaining the same number of experts in every layer. Allowing layer-specific pruning budgets could provide further improvements though selecting these layer-specific pruning budgets remain an interesting research direction. Although our continuous optimization based formulation avoids the combinatorial subset enumeration required by exact reconstruction-based pruning, expert selection and the subsequent weight-update stage still require per-layer optimization, introducing additional one-time pruning cost compared with simpler importance-based heuristics though the latter results in worse model utility.

6 Conclusion

We presented a reconstruction-optimized framework for post-training expert pruning in Mixture-of-Experts language models. Rather than ranking experts independently, our approach formulates pruning as joint subset selection and uses a continuous reformulation to make reconstruction-based expert selection scalable to MoE models with large expert pools. We further introduced an inverse-frequency-weighted reconstruction objective and a lightweight reconstruction-based expert-weight update that adapts the retained experts after pruning. Across Qwen3-30B-A3B, Qwen3-235B-A22B, GLM-4.5-Air, and Mixtral-8x7B on mathematical reasoning and code-generation benchmarks, reconstruction-based expert selection consistently improves over existing pruning methods, particularly under aggressive pruning, while the subsequent weight update recovers additional performance. When exhaustive subset selection is feasible, our continuous optimization based approach also achieves comparable performance. Overall, these results support that jointly optimizing the retained expert subset based on reconstruction quality, rather than selecting experts independently using local importance scores, is a promising, scalable approach to MoE compression.

AI USE STATEMENT

In this work, we used generative AI tools for the following tasks with required disclosure: implement methods, clean and reformat dataset. We have not used generative AI tools for other tasks with required disclosure: help develop theoretical models or conceptual frameworks, formulate mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, support qualitative and thematic data analysis, interpret results, and the rest of the required disclosure tasks: generate synthetic data sets, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, assist with translation are not applicable to this work. Additionally, we used generative AI tools for create or edit software code, edit a research paper to improve readability. We have reviewed all AI-assisted work. AI-assisted code was reviewed and extensively tested for correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide detailed information to facilitate reproduction of our results. Section 3 fully specifies the proposed expert-selection and expert-weight-update objectives and optimization procedure. Section 4.1 describes the evaluated models, datasets, calibration and evaluation protocols, pruning configurations, baselines, and computational resources. Appendix C provides additional implementation details, including calibration-set construction and sampling, evaluation settings, baseline implementations, and optimization hyperparameters. Appendix D reports the full experimental results, including standard errors across seeds.

REFERENCES

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. arXiv preprint arXiv:2108.07732, 2021.

Sikai Bai, Haoxi Li, Jie Zhang, Zicong Hong, and Song Guo. Diep: Adaptive mixture-of-experts compression through differentiable expert pruning. Advances in neural information processing systems, 38:56090–56115, 2026.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. arXiv preprint arXiv:2110.14168, 2021.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers), pp. 1280–1297, 2024.

Zican Dong, Han Peng, Peiyu Liu, Xin Zhao, Dong Wu, Feng Xiao, and Zhifeng Wang. Domain-specific pruning of large mixture-of-experts models with few-shot demonstrations. Advances in Neural Information Processing Systems, 38:103552–103577, 2026.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. Glam: Efficient scaling of language models with mixture-of-experts. In International conference on machine learning, pp. 5547–5569. PMLR, 2022.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. Journal of Machine Learning Research, 23(120):1–39, 2022.

Elias Frantar and Dan Alistarh. Optimal brain compression: A framework for accurate post-training quantization and pruning. Advances in Neural Information Processing Systems, 35:4475–4488, 2022.

Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In International conference on machine learning, pp. 10323–10337. PMLR, 2023.

Shwai He, Daize Dong, Liang Ding, and Ang Li. Towards efficient mixture of experts: A holistic study of compression techniques. arXiv preprint arXiv:2406.02500, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. arXiv preprint arXiv:2103.03874, 2021.

Wentao Hu, Mingkuan Zhao, Shuangyong Song, Xiaoyan Zhu, Xin Lai, and Jiayin Wang. Mosaic pruning: A hierarchical framework for generalizable pruning of mixture-of-experts models. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 40, pp. 21885–21893, 2026.

Weizhong Huang, Yuxin Zhang, Xiawu Zheng, Fei Chao, Rongrong Ji, and Liujuan Cao. Discovering important experts for mixture-of-experts models pruning through a theoretical perspective. Advances in Neural Information Processing Systems, 38:135973–136003, 2026.

Hugging Face. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL `https://github.com/huggingface/open-r1`.

Naman Jain, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In International Conference on Learning Representations, volume 2025, pp. 58791–58831, 2025.

Ajay Jaiswal, Jianyu Wang, Yixiao Li, Pingzhi Li, Tianlong Chen, Zhangyang Wang, Chong Wang, Ruoming Pang, and Xianzhi Du. Finding fantastic experts in moes: A unified study for expert dropping strategies and observations. arXiv preprint arXiv:2504.05586, 2025.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. arXiv preprint arXiv:2401.04088, 2024.

Mike Lasby, Ivan Lazarevich, Nish Sinnadurai, Sean Lie, Yani Ioannou, and Vithursan Thangarasa. Reap the experts: Why pruning prevails for one-shot moe compression. In International Conference on Learning Representations, volume 2026, pp. 146883–146912, 2026.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. {GS}hard: Scaling giant models with conditional computation and automatic sharding. In International Conference on Learning Representations, 2021. URL `https://openreview.net/forum?id=qrwe7XHTmYb`.

Pingzhi Li, Zhenyu Zhang, Prateek Yadav, Yi-Lin Sung, Yu Cheng, Mohit Bansal, and Tianlong Chen. Merge, then compress: Demystify efficient smoe with hints from its routing policy. In International Conference on Learning Representations, volume 2024, pp. 14234–14256, 2024.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In International Conference on Learning Representations, volume 2024, pp. 39578–39601, 2024.

Enshu Liu, Junyi Zhu, Zinan Lin, Xuefei Ning, Matthew B Blaschko, Shengen Yan, Guohao Dai, Huazhong Yang, and Yu Wang. Efficient expert pruning for sparse mixture-of-experts language models: Enhancing performance and reducing inference costs. arXiv preprint arXiv:2407.00945, 2024.

Zongfang Liu, Shengkun Tang, Boyang Sun, Zhiqiang Shen, and Xin Yuan. EvoESAP: Non-uniform expert pruning for sparse MoE. arXiv preprint arXiv:2603.06003, 2026.

Xudong Lu, Qi Liu, Yuhui Xu, Aojun Zhou, Siyuan Huang, Bo Zhang, Junchi Yan, and Hongsheng Li. Not all experts are equal: Efficient expert pruning and skipping for mixture-of-experts large language models. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 6159–6172, 2024.

Ryan Lucas, Kayhan Behdin, Zhipeng Wang, Qingquan Song, Rahul Mazumder, et al. Reasoning models can be accurately pruned via chain-of-thought reconstruction. In International Conference on Learning Representations, volume 2026, pp. 142840–142852, 2026.

Xiang Meng, Kayhan Behdin, Haoyue Wang, and Rahul Mazumder. Alps: Improved optimization for highly sparse one-shot pruning for large language models. Advances in neural information processing systems, 37:37594–37625, 2024a.

Xiang Meng, Kayhan Behdin, Haoyue Wang, and Rahul Mazumder. Alps: Improved optimization for highly sparse one-shot pruning for large language models. Advances in neural information processing systems, 37:37594–37625, 2024b.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, et al. Olmoe: Open mixture-of-experts language models. In International Conference on Learning Representations, volume 2025, pp. 62061–62121, 2025.

Alexandre Muzio, Alex Sun, and Churan He. Seer-moe: Sparse expert efficiency through regularization for mixture-of-experts. arXiv preprint arXiv:2404.05089, 2024.

Guilherme Penedo, Anton Lozhkov, Hynek Kydlíček, Loubna Ben Allal, Edward Beeching, Agustín Piqueres Lajarín, Quentin Gallouédec, Nathan Habib, Lewis Tunstall, and Leandro von Werra. Codeforces. https://huggingface.co/datasets/open-r1/codeforces, 2025.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In International Conference on Learning Representations, 2017.

Mingjie Sun, Zhuang Liu, Anna Bair, and Zico Kolter. A simple and effective pruning approach for large language models. In International Conference on Learning Representations, volume 2024, pp. 4942–4964, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. arXiv preprint arXiv:2505.09388, 2025.

Cheng Yang, Yang Sui, Jinqi Xiao, Lingyi Huang, Yu Gong, Yuanlin Duan, Wenqi Jia, Miao Yin, Yu Cheng, and Bo Yuan. Moe-i2: Compressing mixture of experts models through inter-expert pruning and intra-expert low-rank decomposition. In Findings of the Association for Computational Linguistics: EMNLP 2024, pp. 10456–10466, 2024.

Seohee Yoon and Yong Suk Choi. Compel: Compensated mixture-of-experts pruning with expert-layer distribution. In Findings of the Association for Computational Linguistics: ACL 2026, pp. 30408–30423, 2026.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. arXiv preprint arXiv:2508.06471, 2025.

Zeliang Zhang, Xiaodong Liu, Hao Cheng, Chenliang Xu, and Jianfeng Gao. Diversifying the expert knowledge for task-agnostic pruning in sparse mixture-of-experts. In Findings of the Association for Computational Linguistics: ACL 2025, pp. 86–102, 2025.

Yixiao Zhou, Ziyu Zhao, Dongzhou Cheng, Zhiliang Wu, Jie Gui, Yi Yang, Fei Wu, Yu Cheng, and Hehe Fan. Dropping experts, recombining neurons: Retraining-free pruning for sparse mixture-of-experts LLMs. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), Findings of the Association for Computational Linguistics: EMNLP 2025, pp. 15169–15186, Suzhou, China, November 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-emnlp.820. URL https://aclanthology.org/2025.findings-emnlp.820/.

A Theoretical Motivation for Weighted Reconstruction

We provide an expert-balanced interpretation of the proposed weighted reconstruction objective.

Consider an MoE layer with E experts and a calibration set containing $N = \sum_{m=1}^{M} T_m$ tokens. Let $\mathcal{T}(t)$ denote the set of K experts activated by token t in the original model, where $|\mathcal{T}(t)| = K$. For expert e, define

$$\mathcal{X}_e = \{t : e \in \mathcal{T}(t)\}, \qquad n_e = |\mathcal{X}_e|. \tag{9}$$

Thus, n_e denotes the activation frequency of expert e on the calibration set. We assume $n_e > 0$ for every expert considered below, so that the inverse-frequency weights are well-defined.

To account for the per-sequence length normalization used in Eqs. (5) and (7), let

$$\ell_t(\mathbf{z}) := \frac{N}{2MT_{m(t)}} \mathcal{L}\big(y_t, \widetilde{y}_t(\mathbf{z})\big), \tag{10}$$

where $m(t)$ denotes the calibration sequence containing token t. With this definition, the unweighted reconstruction objective can be written exactly as

$$L_{\text{unweighted}} = \frac{1}{N} \sum_{t=1}^{N} \ell_t. \tag{11}$$

For simplicity, we suppress the dependence of ℓ_t on $\mathbf{z}$ below. Define the expert-conditioned reconstruction loss as

$$L_e = \frac{1}{n_e} \sum_{t \in \mathcal{X}_e} \ell_t. \tag{12}$$

Importantly, L_e is not the reconstruction error attributable solely to expert e. Rather, it is the average sequence-normalized reconstruction loss over calibration tokens for which expert e is activated.

A.1 From Token Averaging to Expert-Balanced Reconstruction

The following proposition characterizes how inverse-frequency token weighting changes the aggregation of expert-conditioned reconstruction losses.

Proposition 1 (Expert-balanced interpretation of inverse-frequency weighting)**.** *Let $p_e := \frac{n_e}{KN}$. Then $\sum_{e=1}^{E} p_e = 1$, and the unweighted reconstruction objective can be written as*

$$L_{\text{unweighted}} = \sum_{e=1}^{E} p_e L_e. \tag{13}$$

Thus, the unweighted objective aggregates expert-conditioned losses in proportion to their calibration-time activation frequencies.

In contrast, for the proposed token weight $w_t = \sum_{e \in \mathcal{T}(t)} \frac{1}{n_e}$, the weighted reconstruction objective satisfies

$$L_{\text{weighted}} := \frac{1}{N} \sum_{t=1}^{N} w_t \ell_t = \frac{1}{N} \sum_{e=1}^{E} L_e. \tag{14}$$

Consequently, up to the constant factor E/N, minimizing the weighted objective is equivalent to minimizing the uniform expert average $\frac{1}{E} \sum_{e=1}^{E} L_e$.

Proof. Since every token activates exactly K experts,

$$\sum_{e=1}^{E} \sum_{t \in \mathcal{X}_e} \ell_t = K \sum_{t=1}^{N} \ell_t. \tag{15}$$

Therefore,

$$L_{\text{unweighted}} = \frac{1}{N} \sum_{t=1}^{N} \ell_t = \frac{1}{KN} \sum_{e=1}^{E} \sum_{t \in \mathcal{X}_e} \ell_t = \sum_{e=1}^{E} \frac{n_e}{KN} L_e = \sum_{e=1}^{E} p_e L_e. \tag{16}$$

For the weighted objective,

$$\sum_{t=1}^{N} w_t \ell_t = \sum_{t=1}^{N} \sum_{e \in \mathcal{T}(t)} \frac{1}{n_e} \ell_t = \sum_{e=1}^{E} \frac{1}{n_e} \sum_{t \in \mathcal{X}_e} \ell_t = \sum_{e=1}^{E} L_e. \tag{17}$$

Dividing by N gives Eq. (14). Since E/N is independent of the selection variables, minimizing this objective is equivalent to minimizing the uniform expert average defined above. □

Proposition 1 gives a direct interpretation of the proposed weighting. Although the standard reconstruction objective assigns the prescribed sequence-normalized weight to each token, when viewed through expert-conditioned losses it weights expert e proportionally to its activation frequency p_e. The proposed inverse-frequency weighting removes this frequency dependence and gives every expert-conditioned loss equal relative weight.

A.2 A Frequency-Independent Bound on Expert-Conditioned Loss

The expert-balanced formulation also yields a simple consequence. Since $L_e \geq 0$ for every expert,

$$L_e \leq \sum_{j=1}^{E} L_j = E \left(\frac{1}{E} \sum_{j=1}^{E} L_j \right), \qquad \forall e. \tag{18}$$

Hence,

$$\max_e L_e \leq E L_{\text{bal}}, \qquad L_{\text{bal}} := \frac{1}{E} \sum_{e=1}^{E} L_e. \tag{19}$$

By comparison, Eq. (13) gives

$$p_e L_e \leq L_{\text{unweighted}}, \tag{20}$$

and therefore

$$L_e \leq \frac{1}{p_e} L_{\text{unweighted}} = \frac{KN}{n_e} L_{\text{unweighted}}. \tag{21}$$

Thus, at the level of these elementary bounds, the bound associated with the expert-balanced objective does not explicitly deteriorate with expert activation frequency, whereas the corresponding bound obtained from the unweighted objective scales as $1/p_e$.

A.3 Robustness to Reweighting of Expert-Conditioned Losses

A related interpretation follows by considering different relative weights on the expert-conditioned losses. Let

$$\mathbf{q} = (q_1, \ldots, q_E), \qquad q_e \geq 0, \qquad \sum_{e=1}^{E} q_e = 1,$$

and define

$$L_{\mathbf{q}} = \sum_{e=1}^{E} q_e L_e. \tag{22}$$

For the unweighted objective,

$$L_{\mathbf{q}} = \sum_{e=1}^{E} \frac{q_e}{p_e} p_e L_e \leq \left(\max_e \frac{q_e}{p_e} \right) L_{\text{unweighted}}. \tag{23}$$

This bound can become loose when an expert has small calibration frequency p_e but receives substantially larger relative weight under $\mathbf{q}$.

For the expert-balanced objective,

$$L_{\mathbf{q}} \leq \max_e L_e \leq E L_{\text{bal}}. \tag{24}$$

Equivalently,

$$\sup_{\mathbf{q} \in \Delta_E} L_{\mathbf{q}} = \max_e L_e \leq E L_{\text{bal}}, \tag{25}$$

where Δ_E denotes the probability simplex over experts.

Thus, for fixed expert-conditioned losses $\{L_e\}$, the expert-balanced formulation provides a bound that does not depend on the calibration frequencies $\{p_e\}$. This gives a robustness interpretation with respect to changing the relative weights assigned to expert-conditioned losses. We emphasize that this argument considers reweighting of fixed $\{L_e\}$ and does not by itself characterize general distribution shift, under which the expert-conditioned losses may also change.

Connection to frequency-normalized expert pruning The above motivation is related to the frequency-normalization principle used in prior MoE pruning methods such as REAP. Such methods seek to avoid conflating an expert's functional importance with how frequently it is activated. Our weighting applies a similar principle directly to the reconstruction objective: instead of allowing frequently activated experts to dominate token-level reconstruction, we normalize their contribution such that each expert-conditioned reconstruction loss receives equal weight.

B Joint Reconstruction Optimization versus Fixed-Score Expert Selection

In this section, we connect our reconstruction-based optimization to importance-score expert pruning. We show that a local first-order approximation of the OptEP objective reduces exactly to a fixed-score top-s expert-selection rule. The resulting score is the local negative gradient of the reconstruction objective with respect to each expert-selection variable. The full OptEP optimization can therefore be viewed as generalizing a fixed-score rule by continually updating these local scores as the reconstructed mixture changes.

B.1 Local Reconstruction Scores

For notational simplicity, we flatten the calibration sequences into tokens indexed by t. Let x_t denote the layer input activation of token t, and define

$$a_{t,e} = a_e(x_t), \qquad f_{t,e} = f_e(x_t;\theta_e), \qquad y_t = y(x_t).$$

Since the pretrained router and expert weights Θ are fixed during expert selection, we write

$$\widetilde{y}_t(\mathbf{z}) := \widetilde{y}(x_t;\mathbf{z},\Theta) = \frac{\sum_{e=1}^{E} z_e a_{t,e} f_{t,e}}{\sum_{e=1}^{E} z_e a_{t,e}}. \tag{26}$$

Define

$$D_t(\mathbf{z}) = \sum_{j=1}^{E} z_j a_{t,j}, \tag{27}$$

and assume $D_t(\mathbf{z}) > 0$ for the selection vectors considered below.

Let c_t denote the coefficient of token t in the continuous reconstruction objective in Eq. (5); specifically, $c_t = 1/(2MT_{m(t)})$ for a token belonging to calibration sequence $m(t)$. The reconstruction objective can therefore be written as

$$R(\mathbf{z}) = \sum_t c_t \mathcal{L}\big(y_t, \widetilde{y}_t(\mathbf{z})\big). \tag{28}$$

Differentiating the normalized surrogate output with respect to z_e gives

$$\frac{\partial \widetilde{y}_t(\mathbf{z})}{\partial z_e} = \frac{a_{t,e} f_{t,e} D_t(\mathbf{z}) - a_{t,e} \sum_j z_j a_{t,j} f_{t,j}}{D_t(\mathbf{z})^2} = \frac{a_{t,e}}{D_t(\mathbf{z})} \left(f_{t,e} - \widetilde{y}_t(\mathbf{z})\right). \tag{29}$$

Consequently, wherever $\mathcal{L}$ is differentiable,

$$\frac{\partial R(\mathbf{z})}{\partial z_e} = \sum_t c_t \frac{a_{t,e}}{D_t(\mathbf{z})} \left\langle \nabla_2 \mathcal{L}\big(y_t, \widetilde{y}_t(\mathbf{z})\big), f_{t,e} - \widetilde{y}_t(\mathbf{z}) \right\rangle. \tag{30}$$

Equation (30) gives a local importance score for each expert under the continuous reconstruction objective. In particular, the local marginal effect of expert e depends not only on its own router score and output, but also on the current reconstructed mixture through $D_t(\mathbf{z})$ and $\widetilde{y}_t(\mathbf{z})$.

For the squared reconstruction loss

$$\mathcal{L}(y, \widetilde{y}) = \|y - \widetilde{y}\|_2^2,$$

define the local marginal reconstruction score

$$S_e(\mathbf{z}) := -\frac{\partial R(\mathbf{z})}{\partial z_e}. \tag{31}$$

Using Eq. (30), we obtain

$$S_e(\mathbf{z}) = \frac{1}{M} \sum_t \frac{1}{T_{m(t)}} \frac{a_{t,e}}{D_t(\mathbf{z})} \langle y_t - \widetilde{y}_t(\mathbf{z}), f_{t,e} - \widetilde{y}_t(\mathbf{z}) \rangle . \tag{32}$$

The inner product in Eq. (32) compares the current reconstruction residual, $y_t - \widetilde{y}_t(\mathbf{z})$, with the direction in which increasing z_e moves the reconstructed mixture, $f_{t,e} - \widetilde{y}_t(\mathbf{z})$. Thus, an expert has a large positive local score when increasing its selection weight moves the current mixture in a direction that reduces the reconstruction residual.

B.2 First-Order Approximation Yields Fixed-Score Selection

The connection to fixed-score pruning can be made explicit by locally linearizing the reconstruction objective. Consider a reference point $\mathbf{z}^{(0)}$ at which R is differentiable and $D_t(\mathbf{z}^{(0)}) > 0$ for every calibration token. A first-order Taylor expansion gives

$$R(\mathbf{z}) = R(\mathbf{z}^{(0)}) + \nabla R(\mathbf{z}^{(0)})^\top (\mathbf{z} - \mathbf{z}^{(0)}) + o\big(\|\mathbf{z} - \mathbf{z}^{(0)}\|\big). \tag{33}$$

Define the local expert score

$$s_e^{(0)} := -\left. \frac{\partial R(\mathbf{z})}{\partial z_e} \right|_{\mathbf{z}=\mathbf{z}^{(0)}}. \tag{34}$$

Ignoring terms independent of $\mathbf{z}$ and higher-order terms, the local linear model becomes

$$R_{\text{lin}}(\mathbf{z}) = -\sum_{e=1}^{E} s_e^{(0)} z_e + \text{const}. \tag{35}$$

Proposition 2 (First-order approximation yields fixed-score selection)**.** *Let $\mathbf{z}^{(0)}$ be a reference point satisfying the conditions above. Minimizing the first-order approximation in Eq.* (35) *subject to*

$$\mathbf{z} \in \{0,1\}^E, \qquad \sum_{e=1}^{E} z_e = s,$$

is exactly equivalent to retaining the s experts with the largest local scores $s_e^{(0)}$.

Interpretation as a frozen-gradient pruning rule Proposition 2 shows that fixed-score expert selection arises naturally as a local linearization of our reconstruction objective. At a reference mixture $\mathbf{z}^{(0)}$, the score $s_e^{(0)}$ measures the instantaneous decrease in reconstruction error associated with increasing the selection weight of expert e. If these scores are computed once and then frozen, expert selection reduces exactly to a standard top-s importance-ranking rule.

This provides a structural connection between reconstruction optimization and score-based pruning. The first-order model replaces the nonlinear reconstruction objective by a single additive expert score computed at one reference mixture.

B.3 Dynamic Scores under Joint Reconstruction Optimization

The full OptEP optimization differs from the first-order approximation because it does not freeze the expert scores at a single reference point. Instead, the local score at the current iterate is

$$S_e(\mathbf{z}) = -\frac{\partial R(\mathbf{z})}{\partial z_e}, \tag{36}$$

which changes as $\mathbf{z}$ changes.

For the squared reconstruction loss, Eq. (32) makes this dependence explicit: both the normalization factor $D_t(\mathbf{z})$ and the reconstructed mixture $\widetilde{y}_t(\mathbf{z})$ depend on the current values of all expert-selection variables. As optimization changes $\mathbf{z}$, the reconstruction residual changes, and the marginal value of each expert is therefore recomputed relative to the updated mixture.

This captures interactions that are absent from a frozen-score rule. For example, two experts may both receive large local scores at an initial reference point because each helps reduce the same component of the reconstruction residual. After one of them receives a larger selection weight, that component of the residual may already be substantially reconstructed, reducing the marginal value of the other expert. Conversely, an expert with a smaller initial score may become relatively more valuable if it provides a complementary direction that addresses the residual left by the currently emphasized experts.

Thus, the distinction between fixed-score selection and OptEP can be viewed as the distinction between a single frozen local ranking and a selection-dependent ranking that evolves throughout nonlinear optimization.

Connection to existing score-based pruning Methods such as Frequency, EASY-EP, and REAP compute a scalar importance score for each expert and retain the highest-scoring experts. Although they differ in how the expert scores are constructed, their final selection rule has the additive form

$$\max_{\mathbf{z}} \sum_{e=1}^{E} s_e z_e \qquad \text{subject to} \qquad \mathbf{z} \in \{0,1\}^E, \quad \sum_{e=1}^{E} z_e = s. \tag{37}$$

The first-order approximation of OptEP has the same fixed-score selection structure, but with scores derived from the local reconstruction gradient. The full nonlinear continuous reconstruction optimization goes beyond this static approximation by allowing the effective expert scores to change as the reconstructed mixture evolves.

C Experiments Details

C.1 Calibration Data Details

Qwen/GLM calibration data For the Qwen and GLM models, we construct a separate calibration set for each of the three seeds by randomly sampling prompt–completion pairs until reaching the fixed budget of $2^{19} = 524{,}288$ tokens. Calibration samples for coding are drawn from Codeforces problems and calibration samples for math are drawn from OpenR1-Math-10k problems, together with their corresponding model-generated completions, including reasoning traces. We limit the calibration samples to correct only responses. We generate these completions using temperature 0.6, top-p 0.95, and a maximum of 32,768 new tokens in both domains. Depending on the seed, the fixed token budget corresponds to 48 coding samples for Qwen3-30B-A3B, 43–46 coding samples for Qwen3-235B-A22B, and 75–93 coding samples for GLM-4.5-Air; for math, it corresponds to 77 samples for Qwen3-30B-A3B, 63 samples for Qwen3-235B-A22B, and 49–58 samples for GLM-4.5-Air.

Mixtral-8x7B calibration data Mixtral-8x7B performs poorly on MATH-500 and LiveCodeBench, so we use easier math and coding benchmarks for both calibration and evaluation. Unlike the Qwen setting, where calibration and evaluation use different datasets, Mixtral-8x7B uses different splits of the same dataset: GSM8K for math and MBPP for coding.

For math, the Mixtral-8x7B calibration pool consists of all 7,473 GSM8K training examples together with completions generated by the full model. We generate completions using temperature 0.0, top-p 1.0, and a maximum of 512 new tokens for both math and coding. For each of five seeds, we randomly sample prompt–completion pairs with correct full-model responses until reaching a fixed budget of 262,144 tokens. This results in 1,248–1,276 GSM8K training examples per calibration set, depending on the seed.

For coding, we use the full MBPP dataset of 974 examples to construct seed-specific calibration and evaluation splits. For each seed, we generate full-model completions using the same decoding settings and randomly sample prompt–completion pairs until reaching the 262,144-token calibration budget. We then retain only traces for which the full model produces the correct answer, resulting in 267–287 calibration sequences per seed. Examples sampled toward the token budget but yielding incorrect responses are discarded and used in neither calibration nor evaluation. Excluding these examples

from evaluation avoids artificially increasing the proportion of incorrect full-model responses in the evaluation set. The remaining MBPP examples form the corresponding evaluation split (details below).

C.2 Evaluation Details

Qwen/GLM evaluation We evaluate the Qwen and GLM pruned models zero-shot on 500 problems in MATH-500 and 268 problems in LiveCodeBench (LCB v4–v5), using a single generation per problem with temperature 0.6, top-$p = 0.95$, and a maximum generation length of 32,768 tokens. We report mean pass@1 accuracy and standard error across three seeds, where each seed determines both the sampled calibration set and the stochastic model generations during evaluation. Thus, the reported standard error reflects variability from both calibration-set selection and inference-time sampling.

Mixtral-8x7B evaluation For Mixtral-8x7B math, we evaluate mathematical reasoning on the full GSM8K test set of 1,319 questions. We use a single greedy generation per problem (temperature 0) with a maximum generation length of 512 tokens and report mean pass@1 accuracy and standard error across five calibration seeds. Because decoding is deterministic, there is no additional inference-time sampling variability. For GSM8K, the standard error therefore reflects only variation in the sampled calibration set.

For Mixtral-8x7B coding, we evaluate code generation on a held-out MBPP set that is disjoint from the calibration set. MBPP is therefore the only setting in which the evaluation set varies across seeds as well as calibration, since the dataset is not large enough to reserve a separate fixed evaluation set while maintaining the calibration budget. MBPP is evaluated on 477–485 problems per seed rather than a fixed test split. For each seed, the calibration subset and its complementary evaluation subset are fixed and shared across all compared methods. For MBPP evaluation the reported standard error reflects variation in both the calibration and evaluation subsets, since the calibration seed determines the corresponding held-out evaluation set.

C.3 Baselines

We provide additional details on the baselines used:

- **Frequency pruning** (Muzio et al., 2024) counts the number of tokens for which each expert is activated and retains the experts with the highest activation counts in each layer.
- **Random pruning** randomly selects the subset of experts to retain independently in each layer.
- **EASY-EP** (Dong et al., 2026) assigns each expert an importance score based on its router probability, expert output norm, and token-importance weight, where token importance is computed from the cosine similarity between the MoE-layer input and output. Experts with the highest scores are retained.
- **REAP** (Lasby et al., 2026) scores each expert using its router weight and activation magnitude, averaged over the calibration tokens for which that expert is active. This reduces the influence of activation frequency and helps retain infrequently used experts that make large contributions when selected. Experts with the highest scores are retained.
- **DiEP** (Bai et al., 2026) uses a differentiable, non-uniform pruning strategy in which each expert and each layer are assigned learned importance scores. The expert- and layer-level scores are jointly trained and combined to determine which experts to prune, allowing different numbers of experts to be retained across layers.
- **M-SMoE** (Li et al., 2024) combines expert pruning with expert merging. Rather than simply dropping the pruned experts, it merges experts based on the routing structure, producing a compressed set of experts with updated weights.
- **NAEE** (Lu et al., 2024) performs layerwise reconstruction-based expert pruning. For each layer, it selects the subset of experts that minimizes the reconstruction error between the original and pruned MoE-layer outputs. The optimization is solved by enumerating all candidate expert subsets, making it practical primarily for MoEs with relatively few experts per layer. Because the Qwen models contain 128 experts per layer, exhaustive enumeration is computationally infeasible. We therefore omit this method for Qwen and evaluate it only on Mixtral-8x7B, which has 8 experts per layer.

C.4 Optimization Details

Pruning-only optimization OptEP (uw), OptEP (w) and OptEP (w-alt) all optimize expert selection independently for each layer. For every layer, the optimization uses activations from a forward pass

of the full, unpruned model; pruning decisions from earlier layers are therefore not propagated to subsequent layers. We use this independent layerwise setup to ensure a fair comparison with the pruning baselines, which likewise select experts independently across layers.

For the expert-selection step in Eq. (5), we optimize each layer for 50 steps using SGD without momentum and a learning rate of 0.02. We initialize $z_e = 0.5$ for all experts, clip the gradient norm to 5, and clamp the selection variables to $[0, 1]$ after each update. The pretrained router and expert weights remain fixed throughout this stage. We use $\alpha = 1$ for Mixtral-8x7B and $\alpha = 2$ for both Qwen and GLM models.

Weight update optimization For the reconstruction-based weight update in Eq. (8), we optimize only the retained expert weights while keeping the router and all other parameters of the pruned model fixed. Unlike the pruning-only stage, weight updates are performed sequentially across layers: after updating a layer, we run the partially updated pruned model forward to generate the inputs used to optimize the next layer.

For Qwen and GLM, we use Adam with a learning rate of 10^{-4}, gradient-norm clipping at 1, and $\lambda = 0.01$, using the same calibration data as in the pruning stage without a separate validation set. Because the Qwen and GLM calibration datasets contain relatively few samples, we do not hold out a separate validation set to select the number of optimization steps. Instead, we fix the number of steps to either 25 or 50, depending on the setting. We use 50 optimization steps per layer for the smaller Qwen3-30B-A3B model and 25 steps per layer for the larger Qwen3-235B-A22B and GLM-4.5-Air models.

For Mixtral-8x7B, we optimize the retained expert weights for up to 150 steps using Adam with a learning rate of 3×10^{-4}, gradient-norm clipping at 1, and $\lambda = 0.01$. We split the calibration data into 80% training and 20% validation subsets, evaluate validation loss every 10 optimization steps, and select the checkpoint with the lowest validation loss for each layer.

D Additional Results

We provide the full results for all evaluated models in Tables 5, 6, 7, 8, 9, 10, and 11. These tables report mean performance together with standard errors across experimental seeds and include results for the additional token-weighting mechanisms considered in our experiments.

D.1 Alternative Weighted Reconstruction Objective: OptEP (w-alt)

In addition to OptEP (w), which aggregates the inverse activation frequencies of all experts active for a token, we consider an alternative weighting scheme that places greater emphasis on whether a token is routed through a particularly infrequently activated expert. For a token with layer input $\boldsymbol{x}$, we define

$$w(\boldsymbol{x}) = \frac{1}{\min_{e \in \mathcal{T}_K(\boldsymbol{a}(\boldsymbol{x}))} n_e}. \tag{38}$$

Thus, the weight is determined by the least frequently activated expert among those active for the token. Whereas OptEP (w) accounts for the inverse frequencies of all active experts, OptEP (w-alt) focuses on the rarest active expert, thereby more directly emphasizing tokens involving infrequently activated experts. The full results in Appendix D also include this weighting mechanism.

Table 5: Expert pruning on Qwen3-30B-A3B, evaluated on MATH-500. Results are mean accuracy ± standard error over three calibration seeds.

Method	Experts retained per layer		
	s=32	s=48	s=64
Random	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Frequency	1.40 ± 0.12	19.27 ± 4.48	66.33 ± 7.80
EASY-EP	31.73 ± 4.68	$\underline{93.73 \pm 0.13}$	94.60 ± 0.40
REAP	76.07 ± 3.21	91.93 ± 0.58	94.73 ± 0.24
M-SMoE	1.33 ± 0.29	1.00 ± 0.31	1.07 ± 0.41
OptEP (uw)	$\underline{88.40 \pm 0.50}$	93.40 ± 0.20	94.67 ± 0.29
OptEP (w)	83.47 ± 1.44	93.53 ± 0.55	95.47 ± 0.35
OptEP (w-alt)	84.47 ± 2.19	93.40 ± 0.31	$\underline{95.67 \pm 0.53}$
OptEP (uw) + EWU	$\mathbf{90.20 \pm 0.53}$	93.27 ± 0.58	95.33 ± 0.41
OptEP (w) + EWU	87.40 ± 2.10	$\mathbf{94.40 \pm 0.12}$	$\mathbf{96.13 \pm 0.07}$
Full model		96.40 ± 0.12	

Table 6: Expert pruning on Qwen3-30B-A3B, evaluated on LiveCodeBench v4–v5. Results are mean accuracy ± standard error over three calibration seeds.

Method	Experts retained per layer		
	s=48	s=64	s=96
Random	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Frequency	0.00 ± 0.00	1.87 ± 0.43	$\mathbf{61.44 \pm 0.25}$
EASY-EP	17.54 ± 1.92	54.73 ± 0.97	61.07 ± 0.69
REAP	38.43 ± 1.68	52.49 ± 0.76	59.83 ± 1.06
M-SMoE	0.00 ± 0.00	0.00 ± 0.00	59.83 ± 0.90
OptEP (uw)	50.50 ± 1.63	56.22 ± 0.82	60.57 ± 1.19
OptEP (w)	48.88 ± 0.37	54.60 ± 1.96	$\mathbf{61.44 \pm 1.43}$
OptEP (w-alt)	48.13 ± 1.63	55.85 ± 1.37	59.08 ± 0.66
OptEP (uw) + EWU	$\underline{52.11 \pm 1.86}$	$\mathbf{57.71 \pm 0.25}$	60.70 ± 0.33
OptEP (w) + EWU	$\mathbf{53.48 \pm 0.69}$	$\underline{56.59 \pm 1.32}$	$\underline{61.19 \pm 0.75}$
Full model		60.57 ± 0.45	

Table 7: Expert pruning on Qwen3-235B-A22B, evaluated on MATH-500. Results are mean accuracy ± standard error over three calibration seeds.

Method	Experts retained per layer		
	s=32	s=48	s=64
Random	0.67 ± 0.29	1.13 ± 0.07	0.60 ± 0.00
Frequency	0.07 ± 0.07	0.27 ± 0.07	93.20 ± 0.46
EASY-EP	71.33 ± 7.64	93.20 ± 0.92	95.20 ± 0.50
REAP	84.07 ± 2.42	93.60 ± 0.23	96.07 ± 0.37
M-SMoE	0.67 ± 0.29	0.13 ± 0.07	60.07 ± 13.75
OptEP (uw)	87.80 ± 0.53	92.53 ± 1.38	95.53 ± 0.29
OptEP (w)	$\underline{91.87 \pm 0.58}$	95.27 ± 0.24	96.13 ± 0.29
OptEP (w-alt)	91.53 ± 0.55	94.53 ± 0.66	$\underline{96.20 \pm 0.53}$
OptEP (uw) + EWU	90.53 ± 1.12	$\underline{95.53 \pm 0.35}$	$\underline{96.20 \pm 0.20}$
OptEP (w) + EWU	$\mathbf{94.67 \pm 0.13}$	$\mathbf{96.13 \pm 0.29}$	$\mathbf{96.47 \pm 0.35}$
Full model		97.60 ± 0.00	

Table 8: Expert pruning on Qwen3-235B-A22B, evaluated on LiveCodeBench v4–v5. Results are mean accuracy ± standard error over three calibration seeds.

Method	Experts retained per layer		
	s=48	s=64	s=96
Random	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Frequency	0.00 ± 0.00	45.52 ± 2.06	64.43 ± 2.37
EASY-EP	41.67 ± 4.21	56.97 ± 0.33	61.19 ± 1.41
REAP	41.17 ± 2.61	61.69 ± 1.73	64.68 ± 1.40
M-SMoE	0.00 ± 0.00	27.11 ± 2.52	64.68 ± 1.76
OptEP (uw)	43.04 ± 2.78	62.31 ± 0.78	65.17 ± 0.66
OptEP (w)	51.99 ± 1.08	62.44 ± 0.82	64.18 ± 1.41
OptEP (w-alt)	51.37 ± 2.16	61.94 ± 0.65	$\mathbf{66.67 \pm 0.90}$
OptEP (uw) + EWU	$\underline{54.85 \pm 1.12}$	$\mathbf{63.31 \pm 0.70}$	65.30 ± 1.14
OptEP (w) + EWU	$\mathbf{56.22 \pm 1.43}$	$\underline{62.94 \pm 2.61}$	$\underline{66.42 \pm 1.51}$
Full model		66.67 ± 0.66	

Table 9: Expert pruning on GLM-4.5-Air, evaluated on MATH-500. Results are mean accuracy ± standard error over three calibration seeds.

Method	Experts retained per layer		
	s=32	s=48	s=64
Random	1.40 ± 0.12	5.13 ± 0.64	27.07 ± 6.89
Frequency	26.47 ± 1.16	56.93 ± 4.91	74.87 ± 3.96
EASY-EP	71.80 ± 2.66	92.07 ± 1.07	95.53 ± 0.79
REAP	75.87 ± 2.03	93.73 ± 0.07	96.60 ± 0.31
M-SMoE	2.47 ± 0.58	27.53 ± 2.49	67.27 ± 3.46
OptEP (uw)	77.73 ± 1.84	92.53 ± 1.09	$\mathbf{97.07 \pm 0.07}$
OptEP (w)	80.47 ± 0.64	93.73 ± 0.07	96.20 ± 0.40
OptEP (w-alt)	77.27 ± 0.66	93.33 ± 0.18	96.47 ± 0.18
OptEP (uw) + EWU	$\underline{85.93 \pm 0.85}$	$\underline{94.20 \pm 0.12}$	$\underline{97.00 \pm 0.20}$
OptEP (w) + EWU	$\mathbf{86.93 \pm 0.98}$	$\mathbf{94.93 \pm 0.24}$	$\underline{97.00 \pm 0.12}$
Full model		97.87 ± 0.58	

Table 10: Expert pruning on GLM-4.5-Air, evaluated on LiveCodeBench v4–v5. Results are mean accuracy ± standard error over three calibration seeds.

Method	Experts retained per layer		
	s=64	s=80	s=96
Random	0.50 ± 0.25	3.48 ± 0.69	22.39 ± 4.93
Frequency	3.24 ± 0.25	14.68 ± 1.51	32.21 ± 2.00
EASY-EP	45.77 ± 1.67	54.85 ± 0.78	$\underline{64.30 \pm 1.08}$
REAP	49.88 ± 0.54	59.33 ± 1.41	64.18 ± 0.37
M-SMoE	1.24 ± 0.12	9.33 ± 0.86	27.49 ± 1.59
OptEP (uw)	51.99 ± 1.39	62.56 ± 2.00	62.06 ± 2.36
OptEP (w)	52.61 ± 0.78	62.69 ± 1.55	$\underline{64.30 \pm 0.69}$
OptEP (w-alt)	53.73 ± 0.43	$\underline{63.06 \pm 0.99}$	$\underline{64.30 \pm 0.12}$
OptEP (uw) + EWU	$\underline{54.85 \pm 0.37}$	$\mathbf{63.56 \pm 0.82}$	$\mathbf{64.31 \pm 0.33}$
OptEP (w) + EWU	$\mathbf{56.22 \pm 1.74}$	61.82 ± 1.00	62.06 ± 1.95
Full model		65.30 ± 0.21	

Table 11: Expert pruning results for Mixtral-8x7B-v0.1 on math (GSM8K flexible extraction) and coding (MBPP standard pass@1). Results are mean accuracy ± standard error in % over five seeds.

Method	Math (GSM8K)		Coding (MBPP)	
	s=4	s=6	s=4	s=6
Random	3.08 ± 1.44	20.15 ± 2.13	7.74 ± 4.68	40.25 ± 0.66
Frequency	8.75 ± 0.48	8.96 ± 0.24	18.71 ± 0.40	37.51 ± 0.60
EASY-EP	7.01 ± 0.24	25.26 ± 0.15	28.70 ± 0.96	47.57 ± 0.34
REAP	11.45 ± 0.79	23.55 ± 0.55	25.07 ± 0.72	44.49 ± 0.21
M-SMoE	10.63 ± 2.33	30.51 ± 0.38	24.45 ± 0.65	38.05 ± 0.38
NAEE brute-force	11.96 ± 0.47	24.96 ± 0.06	28.53 ± 0.47	43.87 ± 0.47
DiEP	10.19 ± 0.27	24.90 ± 0.69	16.54 ± 0.91	41.91 ± 0.60
OptEP (uw)	14.04 ± 0.74	24.58 ± 0.45	27.99 ± 0.63	47.48 ± 0.32
OptEP (w)	12.89 ± 0.00	25.32 ± 0.37	30.31 ± 0.61	47.61 ± 0.59
OptEP (w-alt)	14.54 ± 0.80	22.75 ± 0.33	28.02 ± 0.31	$\underline{48.07 \pm 0.19}$
OptEP (uw) + EWU	$\mathbf{30.42 \pm 0.56}$	$\mathbf{33.48 \pm 0.54}$	$\underline{39.37 \pm 0.77}$	47.61 ± 0.72
OptEP (w) + EWU	$\underline{28.16 \pm 0.54}$	$\underline{32.90 \pm 0.57}$	$\mathbf{39.50 \pm 0.33}$	$\mathbf{50.19 \pm 0.56}$
Brute Force: OptEP (uw) obj	12.37 ± 0.32	21.62 ± 0.44	28.56 ± 0.21	46.40 ± 0.39
Brute Force: OptEP (w) obj	14.63 ± 1.05	23.05 ± 0.54	27.78 ± 0.45	45.86 ± 0.45
Full model (8 experts)	47.61 ± 0.00		59.34 ± 0.86	