MoE Modular Post-Training: Improving and Composing Emergent Domain Experts

Anonymous authors
Paper under double-blind review

Abstract

Modular pretraining enables Mixture-of-Experts (MoE) language models to form domain expert modules that can operate independently. Can these modules also be independently improved and composed? We study *MoE modular post-training*: training domain modules separately and integrating their updates into a shared model. We find that modules fine-tuned with all their parameters can lose much of their improvement when reinserted. Controlled interventions trace this loss not to router mismatch but primarily to co-adaptation between the domain experts and the model's *interface*, the parameters outside the routed experts: both change during fine-tuning, but only the expert updates are reinserted. We therefore introduce *Interface-Frozen Fine-Tuning* (IF-FT), which updates only the domain experts and keeps the interface fixed. Across seven domains on EMO, a 14B-parameter modular MoE, IF-FT preserves standalone accuracy after reinsertion to within test-set noise and improves reinserted accuracy by up to 10 points over full-parameter module fine-tuning. It closely matches training the same experts inside the full model while holding only 3.9B of the 14B parameters in memory during training, and it does not reduce accuracy on other domains. Independently trained modules also compose without further training, and no domain falls below the untrained model; their gains survive best when the modules share few experts. In a standard MoE trained on the same data, reinsertion is equally lossless, but even after training the extracted module falls below its own full model. Emergent domain experts can thus serve as independent units of post-training and composition, provided modularity is built in during pretraining.

1 Introduction

Large language models are trained, updated, and served as monolithic systems: improving one capability means retraining and redeploying the entire model. Mixture-of-Experts (MoE) models (Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2022; Dai et al., 2024) would seem to allow otherwise, since each token activates only a few experts. In standard MoEs, however, experts specialize in tokens and syntactic patterns rather than domains, and restricting a task to a subset of experts breaks the model (Jiang et al., 2024; Muennighoff et al., 2025; Wang et al., 2026a). Pretraining for modularity changes this (Shen et al., 2023; Wang et al., 2026a). We call an MoE *modular* if its pretraining leaves domain-level subsets of experts that work on their own. EMO (Wang et al., 2026a), a recent 14B-parameter example, restricts all tokens of a pretraining document to one shared set of experts; its experts organize by domain, and serving only a domain's top 12.5% of experts costs about three points (we use 25%, as in EMO's fine-tuning test).

Selective deployment is only half of what modularity could offer; the other half concerns post-training. Today, improving an MoE on one domain means loading all of its parameters, updating a model shared by every other domain, and gathering all training data in one place. With a modular MoE one could instead train only a domain's expert subset, on hardware that holds only those experts; reinsert it into the full model without losing the improvement or harming other domains; and compose subsets trained independently, at different times, by different teams, or on data that cannot be pooled. We call this *MoE modular post-training*. On EMO, Wang et al. (2026a) ran a first test: a fine-tuned 32-expert subset, copied back into the 128-expert model, improved the model

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.

but fell short of its own standalone accuracy. Why this *reinsertion gap* arises, and whether it can be avoided, was left open. We therefore ask: *can the expert subsets of a modular MoE be trained apart and plugged back in, alone and together, without losing what they learned, and what determines when this works?*

Existing methods do not settle this question. ESFT (Wang et al., 2024) fine-tunes only task-relevant experts, but inside the full model, which must remain in memory, and it produces no portable unit. Parameter-efficient fine-tuning and model merging (Hu et al., 2022; Ilharco et al., 2023; Yadav et al., 2023) update parameters shared by all tasks, so independently trained updates interfere and need per-combination coefficients. Branch-and-merge methods (Li et al., 2022; Sukhbaatar et al., 2024; Shi et al., 2025) compose separately trained experts, but only along domain partitions fixed before training. None of them tells us whether subsets that *emerge* inside one pretrained MoE can be updated in isolation.

The most natural explanation of the reinsertion gap, that the router no longer recognizes the modified experts, accounts for little of it. We find instead that the experts co-adapt with the rest of the model during fine-tuning. The *interface*, i.e., all parameters other than the routed experts (attention, normalization, embeddings, routers, and the shared expert), through which the routed experts are called and combined, is updated together with the experts, and the experts come to rely on the update. Reinsertion copies back only the experts and discards what they rely on. Repairing the router does not close the gap, whereas two opposite interventions do: keeping the interface fixed during training prevents it, and copying the updated interface into the full model recovers most of it. The effect is strongest on reasoning: on GSM8K, reinsertion keeps only 3.2 of the 19.9 points the expert subset gained.

This diagnosis leads to a simple protocol, *Interface-Frozen Fine-Tuning* (IF-FT; Figure 1d): train only the routed experts of the subset, keep the interface frozen, and write the experts back into their original positions. With IF-FT, reinsertion is lossless within test noise in all seven domains we study, and the resulting model is clearly better than with full-parameter subset fine-tuning in the three reasoning-heavy domains (by 10.0 points on GSM8K) and not worse in the others. Training apart costs little: the result matches training the same experts inside the full model, while holding 3.9B instead of 14B parameters in memory. Independently trained subsets combine by writing each into its own expert positions; their gains survive best when the subsets share few experts, although in our data this is confounded with one domain. Lossless reinsertion also holds for a standard MoE trained on the same data, but there the extracted subset ends up below its own full model, so training it apart is not worthwhile: modularity has to come from pretraining. All experiments use EMO, currently the only public MoE with emergent modularity at this scale.

In summary, our contributions are:

- **Diagnosis.** We trace the reinsertion gap of emergent expert subsets to co-adaptation with a drifted interface rather than to router mismatch, with causal evidence in both directions and a localization to attention and the shared expert (§3.2, §4.3.1).
- **Method.** IF-FT makes reinsertion lossless within test noise across seven domains and matches training the same experts inside the full model while keeping 3.9B instead of 14B parameters in memory (§4.3.2, §4.3.3).
- **Composition.** Independently trained subsets compose by expert slot without pushing any member domain below the untrained model or harming other domains; their gains survive best when the subsets share few experts (§4.3.4).
- **Scope.** We separate what freezing the interface guarantees on the MoEs we test (lossless reinsertion) from what requires modular pretraining (a subset worth training apart) (§4.3.5).

2 Related Work

Specialization and modularity in MoEs. Sparse MoE layers scale capacity at constant per-token compute (Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2022), and recent MoEs combine many fine-grained experts with shared experts (Dai et al., 2024; Muennighoff et al., 2025; Yang et al., 2025). Their experts tend to specialize in tokens and syntax rather than domains (Jiang et al., 2024; Muennighoff et al., 2025; Wang et al., 2026b), so task-specific expert pruning (Chen et al.,

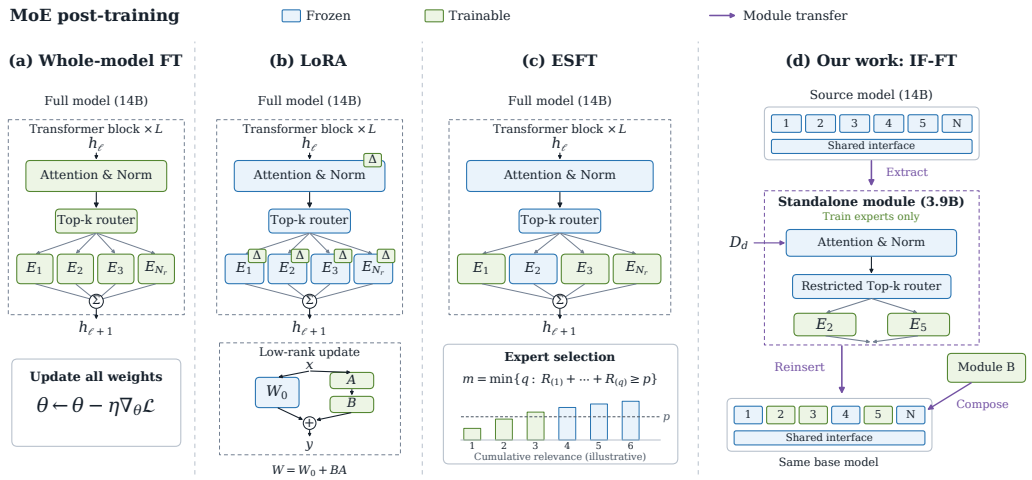


Figure 1: MoE post-training methods. (a–c) Whole-model fine-tuning, LoRA, and ESFT train all weights, low-rank adapters, or threshold-selected experts inside the full 14B model. (d) IF-FT extracts a domain's routed experts and the interface as a standalone 3.9B-parameter module, trains only the experts on domain data D_d, and reinserts them into the unchanged base model, so independently trained modules compose.

2022; Lu et al., 2024; Dong et al., 2025) degrades sharply at small budgets. ModuleFormer (Shen et al., 2023) and EMO (Wang et al., 2026a) instead pretrain MoEs for modularity, and EMO's expert subsets remain usable at 12.5% of the experts. These works address inference; we study how to post-train modular MoEs by training their expert subsets apart and putting them back.

Fine-tuning MoEs. ESFT (Wang et al., 2024) fine-tunes only the experts most relevant to a task, inside the full model; adapters and LoRA (Houlsby et al., 2019; Hu et al., 2022) restrict the update to small or low-rank components (Figure 1a–c). IF-FT shares ESFT's intuition, but trains in an extracted module without the rest of the model in memory, produces a portable set of experts tied to fixed positions, and selects a domain subset that emerged in pretraining.

Composing separately trained components. Branch-Train-Merge, DEMix, BTX, and FlexOlmo train experts on predefined data partitions and combine them into one model (Li et al., 2022; Gururangan et al., 2022; Sukhbaatar et al., 2024; Shi et al., 2025). Weight-space merging combines fine-tuned models or task vectors (Wortsman et al., 2022; Matena & Raffel, 2022; Ilharco et al., 2023; Yadav et al., 2023; Yu et al., 2024), and parameter-efficient modules can be composed arithmetically (Zhang et al., 2023; Huang et al., 2024); these methods act on shared parameters and usually tune coefficients per combination (see also Pfeiffer et al., 2023). We compose subsets that emerge within one pretrained MoE without any predefined partition, and interference can arise only in the expert positions two subsets share, which can be counted before training.

Localizing learned behavior. Restoring or ablating parts of a network has located factual associations in feed-forward layers (Geva et al., 2021; Meng et al., 2022); our diagnosis applies this logic to groups of fine-tuned parameters.

3 Methodology

We first define expert subsets, modules, and the quantities we measure (§3.1), then analyze why the naive protocol can lose its gains (§3.2), which motivates IF-FT (§3.3) and composition by expert slot (§3.4).

3.1 Expert subsets, modules, and reinsertion

Full model and interface. Consider an MoE with L layers. Each layer $l \in \{1, \dots, L\}$ has n_r routed experts $E_{l,1}, \dots, E_{l,n_r}$ and one shared expert $E_l^{(s)}$. Given the hidden state $\boldsymbol{h}_{l,t}$ of token t at layer l, the router computes $\boldsymbol{p}_{l,t} = \mathrm{softmax}(\boldsymbol{W}_l \boldsymbol{h}_{l,t})$, and the layer outputs $\sum_{i \in \mathcal{R}_{l,t}} p_{l,t}(i)\, E_{l,i}(\boldsymbol{h}_{l,t}) + E_l^{(s)}(\boldsymbol{h}_{l,t})$, where $p_{l,t}(i)$ is the i-th entry of $\boldsymbol{p}_{l,t}$ and $\mathcal{R}_{l,t} \subset \{1, \dots, n_r\}$ indexes the k routed experts

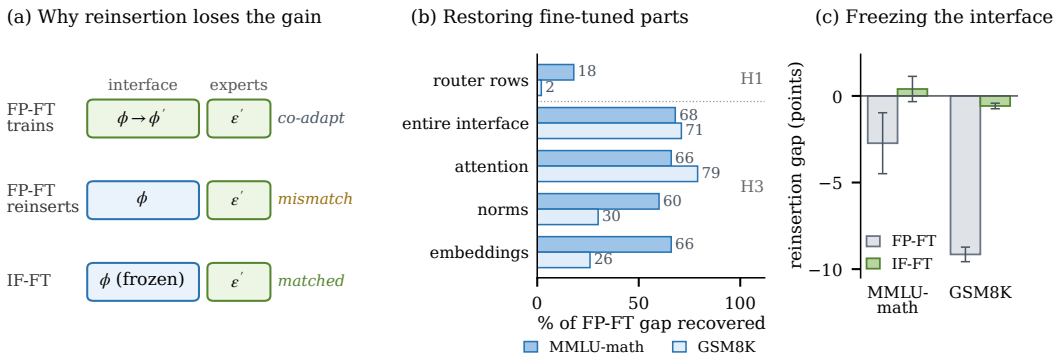


Figure 2: Reinsertion loss comes mainly from expert–interface co-adaptation, and freezing the interface removes it. (a) FP-FT trains the experts ε' while the interface drifts from ϕ to ϕ', but reinsertion pairs ε' with the original ϕ; IF-FT keeps ϕ frozen, so the experts are trained for the interface they are deployed with. (b) Share of the FP-FT reinsertion gap recovered by copying fine-tuned parameters into the reinserted model: router rows (H1) recover at most 18%, the interface (H3) 68–71%. (c) Reinsertion gap (negative means accuracy is lost), mean±sd over three seeds. Panels (b, c) use the short GSM8K schedule (§4.1); exact values and copying details are in Appendix B.

with the highest routing probabilities. We write the parameters as $\theta = (\phi, \varepsilon)$, where $\varepsilon = \{E_{l,i}\}$ are the routed experts and the *interface* ϕ comprises all remaining parameters: the shared experts, attention, normalization layers, embeddings and output head, and routers $\boldsymbol{W}_l$.

Expert subsets, modules, and slots. For a domain with a small calibration set, we select an *expert subset* $\mathcal{S}_l \subset \{1, \ldots, n_r\}$ of m routed experts in each layer, ranking experts by their average routing probability on the calibration tokens, layer by layer (Wang et al., 2026a); other selection methods give nearly the same subsets on EMO (Appendix F). The *module* $M_{\mathcal{S}} = (\phi, \varepsilon_{\mathcal{S}})$ combines the interface with the selected experts $\varepsilon_{\mathcal{S}} = \{E_{l,i} : i \in \mathcal{S}_l\}$ into a standalone model whose routers are restricted to $\mathcal{S}_l$ and renormalized. Following EMO we use $m = 31$ routed experts plus the shared expert (32 of 128), i.e., 3.9B of the full model's 14B parameters. We call the index (l, i) of an expert its *slot*. Our method assumes only that the MoE is *modular*: for each domain there is a subset whose module, without further training, keeps most of the full model's accuracy on that domain.

Training, reinsertion, and metrics. A module is trained on domain data by updating a parameter subset $\psi \subseteq M_{\mathcal{S}}$, which yields experts $\varepsilon'_{\mathcal{S}}$ and possibly an updated interface. *Reinsertion* writes the trained experts back into their slots:

$$\mathrm{R}(\theta, \varepsilon'_{\mathcal{S}}) = \big(\phi,\ \varepsilon'_{\mathcal{S}} \cup \varepsilon_{\neg\mathcal{S}}\big). \tag{1}$$

We measure three quantities. The *reinsertion gap* $g = \mathrm{acc}(\mathrm{R}) - \mathrm{acc}(M')$ compares the reinserted model with the trained module on its own; reinsertion is *lossless* if g is above minus one standard error of the test accuracy. The *gain* $\mathrm{acc}(\mathrm{R}) - \mathrm{acc}(\theta)$ compares it with the untrained model. The *other-domain change* is the mean accuracy change on domains not involved in training. Composition metrics are defined in §3.4.

3.2 Why reinsertion can lose the gain

The protocol of EMO's preliminary test is full-parameter fine-tuning of the module (FP-FT), $\psi = (\phi_M, \varepsilon_{\mathcal{S}})$, where ϕ_M is the module's copy of the interface. FP-FT returns a jointly optimized pair $(\phi'_M, \varepsilon'_{\mathcal{S}})$, but reinsertion (Eq. 1) keeps only $\varepsilon'_{\mathcal{S}}$ and pairs it with the original ϕ. If part of the gain depends on the interface update, that part is lost, and the experts, fitted to ϕ'_M, may misbehave under ϕ (Figure 2a). This is one of three candidate explanations for the gap, each with a testable prediction.

H1 (router mismatch). The routers of the full model, unaware of the modified experts, send them the wrong tokens. *Prediction:* repairing the router, by copying the fine-tuned router rows or retraining the router after reinsertion, closes the gap. **H2 (side effects on other domains).** Modified experts that also belong to other domains' subsets disturb the rest of the model. *Prediction:* other domains degrade after reinsertion. **H3 (interface co-adaptation).** The experts adapted to ϕ'_M and fail under ϕ. *Prediction:* two opposite interventions both close the gap, namely keeping ϕ fixed during training and copying ϕ'_M into the full model together with the experts; the component whose

restoration closes most of the gap is the one the experts depend on. Section 4.3.1 tests these predictions.

3.3 Interface-Frozen Fine-Tuning

If the gap comes from co-adaptation, the remedy is to never let the experts learn an interface that will not be deployed with them. IF-FT trains the module with $\psi = \varepsilon_{\mathcal{S}}$, keeping the entire interface ϕ frozen, including routers and the shared expert, and then reinserts with Eq. 1. The experts are thus optimized against exactly the interface they will meet in the full model (Figure 2a, bottom). The only remaining difference between training and deployment is routing: inside the module each token chooses among 31 routed experts per layer, whereas the full model offers all 127. Section 4.3.3 measures what this restriction costs by training the same experts inside the full model.

IF-FT needs only the module in memory (3.9B parameters, 3.1B of them trainable, instead of 14B), and its output is a set of experts tied to fixed slots of a shared model, which is what makes independently trained modules composable. We do not claim novelty for freezing the parameters outside the routed experts as such; our contribution is to show why the freeze is *necessary* for reinsertion and that it is *sufficient*, i.e., that training in the extracted module loses almost nothing relative to training inside the full model.

3.4 Composition by Expert Slot

Modules trained independently for domains a and b are composed by writing each module's experts into its own slots. Slots in $\mathcal{S}_a \cap \mathcal{S}_b$ receive two versions; by default we average them, and alternatively one domain's version takes priority. Composition requires no training. Because interference can only arise in shared slots, we examine whether the *subset overlap* $\omega_{ab} = \frac{1}{L}\sum_l |\mathcal{S}_{a,l} \cap \mathcal{S}_{b,l}|/m$, computable before any training, indicates how well two modules compose; the number of shared slots is $Lm\,\omega_{ab} = 496\,\omega_{ab}$. We measure composition by the *pooled additivity* and the per-domain *retention*

$$A_{ab} = \frac{G^a_{ab} + G^b_{ab}}{G^a_a + G^b_b}, \qquad r^a_{ab} = \frac{G^a_{ab}}{G^a_a}, \tag{2}$$

where G^x_S is the gain on domain x after inserting the modules in S. $A_{ab} = 1$ means the composed model keeps the full gain of both single modules.

4 Experiments

The experiments answer five questions in the order of our argument: what causes the reinsertion gap (§4.3.1), whether freezing the interface removes it (§4.3.2), what training apart costs (§4.3.3), whether independently trained modules compose (§4.3.4), and which of these results require modular pretraining (§4.3.5).

4.1 Setup

Models. We instantiate modular post-training on EMO-1B-14B trained on 1T tokens (Wang et al., 2026a), to our knowledge the only public MoE with emergent modularity at this scale: 16 layers of 128 experts (127 routed, 1 shared), 8 active per token, 1B active and 14B total parameters. To test what happens when the model is not modular, we use the standard MoE released with EMO, which shares its architecture and training data and differs only in lacking EMO's document-level routing constraint. Appendix I reports a stronger standard MoE, Qwen3-30B-A3B (Yang et al., 2025).

Training protocols. For comparability, all methods share one training recipe: AdamW, peak learning rate 10^{-4} with 10% warmup and linear decay, 3 epochs, batch size 32, and a loss on answer tokens plus a 0.01-weighted load-balancing term. We compare IF-FT with: **FP-FT**, the same module trained with all its parameters; **IF-FT without extraction**, which trains exactly the same experts inside the full model, with native routing over all 127 experts and everything else frozen; **ESFT-Gate** and **ESFT-Token** (Wang et al., 2024), which train, inside the full model, the experts selected by gate scores or token shares up to ESFT's default thresholds (p=0.1 and 0.2); **LoRA** of rank 16, either on the module's experts (merged and reinserted) or on attention and all experts of the full

model; and **whole-model fine-tuning**, for which we report the better of two learning rates. For composition we also train a **joint** baseline that trains the union of two expert subsets inside the full model on both datasets. For GSM8K, the diagnosis (Figure 2) and the comparison with the standard MoE (Table 4) additionally use a *short schedule* (1 epoch, learning rate 5×10^{-5}), on which we ran every intervention and three seeds; §4.4 compares the two schedules. Results are from single runs unless reported as mean±sd over three seeds. Experiments run on nodes of eight NVIDIA H100 GPUs.

4.2 Benchmarks and metrics

We follow EMO's evaluation of selective expert use. MMLU subjects (Hendrycks et al., 2021) are grouped into domains, of which we use math, physics, health, history, psychology, and law; each domain provides a training set, also used as the calibration set for subset selection (e.g., 752 questions for math), and a disjoint test set (e.g., 427). MMLU domains are evaluated 5-shot by ranking answer log-likelihoods (raw accuracy). GSM8K (Cobbe et al., 2021) is trained on its 7,473 training problems and evaluated 8-shot with greedy decoding and exact match on 1,319 test problems, using OLMES prompts (Gu et al., 2024). Standard errors of the test accuracies are 1.9–3.1 points on MMLU domains and 1.2 on GSM8K. The other-domain change is measured on health, physics, history, and psychology, excluding the trained domains (plus MMLU-math for GSM8K). For analysis only, we call math, physics, and GSM8K *reasoning-heavy* and the other domains *knowledge-heavy*. Details are in Appendix A.

4.3 Main results

4.3.1 The reinsertion gap comes from interface co-adaptation

We first ask which of the three explanations accounts for the gap (Figure 2b–c). Repairing the router does not close it: copying the fine-tuned router rows recovers 18% of the gap on math and 2% on GSM8K, and retraining the router after math reinsertion recovers nothing (H1 rejected). After reinserting the FP-FT math experts, the other domains *improve* by 0.9 points on average, so the modified experts do not harm the rest of the model (H2 rejected on math). Both predictions of H3 hold: freezing the interface removes the gap on both tasks, and copying all fine-tuned interface parameters into the reinserted model recovers 68–71% of it.

Restoring one component at a time shows where the dependence resides, and the answer differs by task. On GSM8K, attention recovers 79% of the gap, against 26–30% for norms or embeddings, and the same holds with the full schedule (74%) and on physics (80%). On math, the router rows and shared expert recover about 60% whichever component accompanies them, and attention restored without them recovers 40%. FP-FT therefore learns a *joint* solution in which part of the gain resides in the interface, mainly in attention and, on math, also in the shared expert, and reinsertion discards that part.

4.3.2 Freezing the interface makes reinsertion lossless

Having located the gap in the interface, we ask whether freezing the interface removes it across domains (Table 1). Under FP-FT, the gap is large in all three reasoning-heavy domains (-16.7 on GSM8K, -3.9 on physics, -2.8 on math) and between -0.4 and +2.6 in the four knowledge-heavy ones. Under IF-FT, the gap stays within ± 1.3 points in every domain, inside one standard error: reinsertion is lossless. After reinsertion, IF-FT is clearly better than FP-FT in the reasoning-heavy domains, by 10.0 points on GSM8K and 2.1–3.1 points on physics and math, and within noise of it, never below, in the knowledge-heavy ones.

In knowledge-heavy domains the reinserted model even exceeds the module on its own (e.g., +1.3 on health), since the 96 experts outside the subset remain available. The price is paid by the standalone module, which is weaker under IF-FT in reasoning-heavy domains (40.5 vs. 42.4 on math, 25.8 vs. 31.4 on GSM8K); a module intended only for standalone deployment may still prefer FP-FT. The results are stable: over three seeds, IF-FT reaches 41.8±0.4 on math after reinsertion, an independent rerun of the whole pipeline reproduces the six MMLU domains within 0.8 points (Appendix C), and the other-domain change is positive for every module (+0.4 to +1.9).

Table 1: Reinsertion across seven domains. Accuracy of the trained module on its own and after reinsertion into the full model; gap = reinserted − module. Other-domain change: mean change of the IF-FT model on the other domains. Standard errors: 1.9–3.1 (MMLU), 1.2 (GSM8K).

		FP-FT			IF-FT			Other-domain
Domain	Untrained	module	reinserted	gap	module	reinserted	gap	change
Reasoning-heavy								
GSM8K	11.5	31.4	14.7	−16.7	25.8	24.7	−1.1	+0.4
Physics	42.0	49.0	45.1	−3.9	48.2	48.2	0.0	+0.5
Math	35.8	42.4	39.6	−2.8	40.5	41.7	+1.2	+1.6
Knowledge-heavy								
Health	42.2	45.8	45.8	0.0	46.0	47.3	+1.3	+1.9
History	46.9	46.1	47.7	+1.6	48.0	48.8	+0.8	+0.9
Psychology	53.6	52.5	55.1	+2.6	55.3	55.9	+0.6	+1.2
Law	29.8	30.7	30.3	−0.4	31.4	30.8	−0.6	+0.7

Table 2: Where to train the experts. Accuracy after training with the same recipe (w/o: without); math in parentheses: mean±sd over three seeds. Parameters in billions. Every row updates only routed experts, except FP-FT, which also updates the module's interface; LoRA and whole-model fine-tuning are in Figure 3.

Method (trained in)	Resident	Trainable	Math	Physics	Health	History	GSM8K
Untrained model	—		35.8	42.0	42.2	46.9	11.5
FP-FT (module)	3.9	3.9	39.6	45.1	45.8	47.7	14.7
IF-FT (module)	**3.9**	3.1	41.7 (41.8±0.4)	48.2	47.3	48.8	24.7
IF-FT w/o extraction (full model)	14	3.1	41.7 (42.3±1.1)	49.4	47.8	49.9	26.2
ESFT-Gate (full model)	14	0.13	37.2 (37.0±0.8)	42.0	43.2	46.1	19.3
ESFT-Token (full model)	14	0.23	38.2 (38.0±0.1)	43.2	44.9	46.6	22.2

4.3.3 Training apart costs little

Training in an extracted module restricts routing to 31 experts per layer. We ask what this costs by training the *same* experts inside the full model with the same data and recipe, where tokens can reach all 127 experts (Table 2). On the four MMLU domains, the two agree within 0.0–1.2 points, inside one standard error; with three seeds on math they reach 41.8±0.4 and 42.3±1.1. GSM8K shows the only systematic difference: training without extraction is 1.5 points higher. Restricting routing during training thus costs at most 1.5 points, and only on a reasoning task.

The saving is substantial. Training IF-FT keeps 3.9B instead of 14B parameters in GPU memory (Figure 3), a property of the method rather than of its implementation. It also trains 3.5× faster than the same experts inside the full model at equal batch size (0.45 vs. 0.13 samples/s; Figure 4), although this ratio depends on how the MoE layer is implemented. Every token still activates eight experts either way, so the saving is in memory rather than in computation per token.

Other ways of restricting the update deliver less. With its default thresholds, ESFT trains only 1–2% of the routed experts, because EMO's routing concentrates on a few experts per layer (Appendix F); this yields at most half of IF-FT's gain on MMLU and 59–81% on GSM8K. LoRA with the same recipe also trails IF-FT in all four MMLU domains, e.g., on math 37.9 on the full model and 37.2 on the module's experts, against 41.7 for IF-FT; since LoRA underfits with this recipe, we claim the comparison only for matched training budgets (Appendix E).

4.3.4 Independently trained modules compose by slot

We ask whether independently trained modules can simply be written into the same model. Across ten pairs of MMLU-domain modules, one result is robust: composition never pushes a member domain below the untrained model, and no other domain declines in any pair (other-domain change +0.7 to +2.0). Three modules composed together (math, health, history) also keep all three domains above the untrained model, with $A = 0.72$. How much of the gain survives varies: A ranges from 0.53 to 1.12 and decreases with subset overlap (Spearman ρ between -0.55 and -0.63 depending on the reference single-module gains, $p = 0.05$–0.10). The three pairs with the lowest overlap ($\omega \leq 0.24$) are the only ones with $A \geq 1$, but all three contain history, whose own gain is small (1.6–1.9 points) and benefits from its partner, so overlap and domain identity are confounded. Fixing

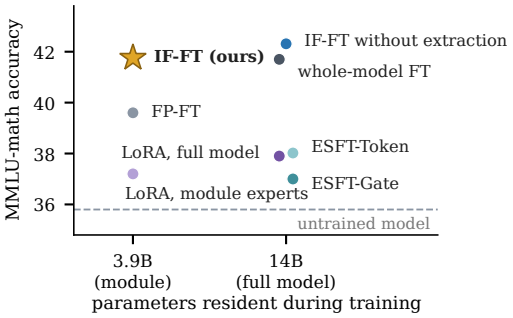


Figure 3: Memory. Accuracy vs. parameters held in GPU memory during training. IF-FT (star) matches the best 14B-resident methods with only 3.9B. Points at 14B are offset for legibility; seed statistics in Table 2.

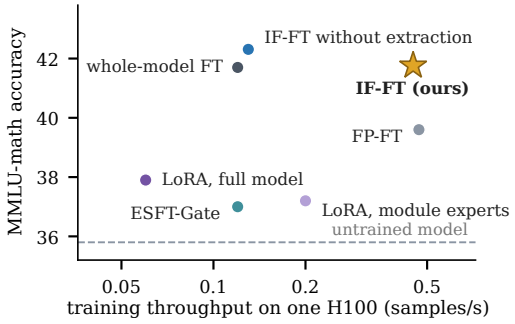


Figure 4: Throughput. Accuracy vs. training throughput on one H100 GPU (batch size 1). IF-FT is 3.5× faster than training the same experts in place; the ratio depends on the implementation (Appendix G).

Table 3: Composing two modules. Accuracy on the two domains (A / B). The three pairs all contain math and span low, medium, and high subset overlap. Pooled additivity: mean±sd over three seeds in parentheses. Joint: union of the two expert subsets trained inside the full model on both datasets.

Subset overlap ω / shared slots	math + history 0.19 / 99	math + health 0.60 / 299	math + physics 0.74 / 369
Untrained model	35.8 / 46.9	35.8 / 42.2	35.8 / 42.0
Each IF-FT module inserted alone	41.7 / 48.8	41.7 / 47.3	41.7 / 48.2
Both IF-FT modules, composed	**41.5 / 49.6**	39.1 / 46.0	39.1 / 45.1
pooled additivity A_{ab}	1.07 (1.09±0.13)	0.64	0.53 (0.70±0.04)
other-domain change	+1.8	+2.0	+0.8
Both LoRA modules (module experts), composed	37.9 / 47.2	37.9 / 43.1	37.9 / 42.8
Joint training (union of subsets, full model)	43.8 / 50.9	42.4 / 47.0	41.5 / 45.5

the measured module reduces this confound (Figure 5): the math and physics modules keep nearly all of their gain when paired with history ($r = 0.97$ and 1.07) and 50–86% with partners of higher overlap, whereas the health module keeps 75–79% with every partner. Subset overlap is thus a useful but imperfect indicator.

Interference sits in the shared slots and can be allocated: giving math priority in the 369 slots it shares with physics raises math's retention from 0.56 to 0.92, at the expense of physics (0.50 to 0.39). Composed LoRA modules trail composed IF-FT modules in every cell (Table 3), largely because their single-domain gains are smaller. Joint training on the union of the two subsets does better still, and composition recovers 62–70% of its gain. At low overlap, joint training captures positive transfer that separate training cannot (math + history: 43.8 / 50.9, above both single modules); at high overlap, it suffers interference itself (physics 45.5, below the physics module alone at 48.2). Unlike composition, joint training also needs all data in one place.

4.3.5 When modular post-training applies

We finally ask which of these results require a modular MoE (Table 4). Lossless reinsertion follows from freezing the interface and holds for the standard MoE as well: its reinserted model is never below its module. Whether the module is worth training apart, however, depends on modularity. Only EMO's module, trained on its own, surpasses its full model (+4.7 on math, +5.0 on GSM8K). The standard MoE's module ends up below its full model (-0.9, -5.3), so its reinserted model improves mainly because the 96 untouched experts fill in, and whole-model fine-tuning serves it better (40.0 vs. 37.5 on math), whereas on EMO the two tie (41.7). Composition shows the same contrast: the standard MoE's math and history subsets share 230 slots instead of 99, and composing them gains +1.6 / -0.3 instead of +5.7 / +2.7. A stronger standard MoE, Qwen3-30B-A3B, shows the same sign (Appendix I). Before any training, the two models already differ in how much accuracy a pruned subset keeps: EMO's 32-expert subset retains 93.5% of the full model's math accuracy and 101% on GSM8K, the standard MoE's only 79.5% and 17%. With two models, whether this retention predicts the value of modular post-training on other MoEs remains a hypothesis.

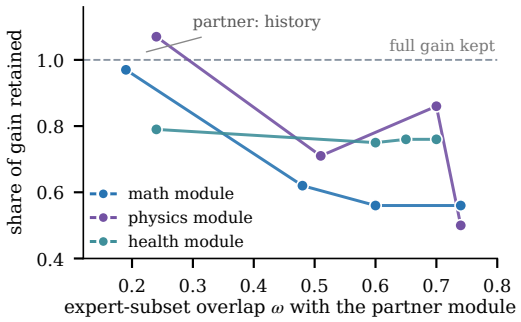


Figure 5: Retention vs. overlap. Share of a module's gain kept after composition, per partner. Math and physics keep nearly all of it with history but 50–86% with other partners; health keeps 75–79%. History and psychology modules (gains within noise) are omitted; see Appendix D.

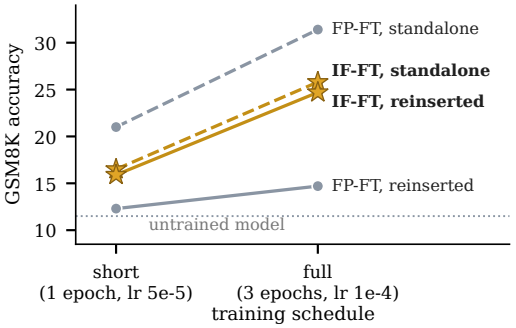


Figure 6: Training length. GSM8K accuracy of the module on its own (dashed) and after reinsertion (solid). Longer training widens the FP-FT gap from −8.7 to −16.7 points; under IF-FT it stays near zero.

Table 4: Is the extracted module worth training? IF-FT on EMO and on the standard MoE trained on the same data (GSM8K with the short schedule). "Module − untrained" compares the trained standalone module with its own untrained full model.

Task	Model	Untrained full model	IF-FT module	Reinserted	Module − untrained
Math	EMO	35.8	40.5	41.7	**+4.7**
	Standard MoE	35.6	34.7	37.5	−0.9
GSM8K	EMO	11.5	16.5	15.9	**+5.0**
	Standard MoE	13.9	8.6	15.5	−5.3

4.4 Ablations and analysis

The gap grows with training. On GSM8K (Figure 6), longer training raises the FP-FT module on its own by 10.4 points but its reinserted version by only 2.4, so the gap doubles from −8.7 to −16.7: the additional gain resides almost entirely in the interface. Under IF-FT, both the module and its reinserted version gain about nine points and the gap stays near zero (−0.6 and −1.1). Math behaves like GSM8K under FP-FT: after 24 training steps reinsertion is lossless (36.8 on its own, 37.0 reinserted), and after 72 steps the gap is −2.8. The advantage of IF-FT thus grows with training.

Reasoning-heavy domains bear the gap. Across Table 1, the FP-FT gap is negative in every reasoning-heavy domain and near zero or positive in every knowledge-heavy one, consistent with factual knowledge residing in feed-forward layers (Geva et al., 2021; Meng et al., 2022), where it travels with the experts, while reasoning also depends on the interface. Because the grouping is post hoc, we treat this as an interpretation.

5 Conclusion

Post-training of a modular MoE can itself be modular, provided the interface is left untouched: the reinsertion gap comes from experts co-adapting with a drifted interface, mainly attention and the shared expert, not mainly from the router. Interface-Frozen Fine-Tuning removes the gap in seven domains, matches training the same experts inside the full model with 3.9B instead of 14B parameters in memory, and yields modules that compose by slot without harming other domains. Lossless reinsertion holds on both MoEs we study, but a module worth training apart requires modularity built in during pretraining.

Limitations and outlook. Our evidence comes from one modular MoE (EMO), MMLU-derived domains, and one generative task; other routes to modularity (Shen et al., 2023; Shi et al., 2025) remain untested. Many results are single runs, the LoRA comparison is budget-matched, and our low-overlap pairs all contain history. Sequential learning and capability removal are natural next steps.

AI USE STATEMENT

We used large language models to assist with writing and polishing the text of this paper. All research ideas, experiments, analyses, and results are the authors' own; the authors reviewed and verified all AI-assisted text and take full responsibility for the content of the paper.

ETHICS STATEMENT

This work fine-tunes and evaluates publicly released models (EMO-1B-14B, the standard MoE released with it, and Qwen3-30B-A3B) on publicly released benchmarks (MMLU and GSM8K); it involves no new human-subject data collection, crowdsourcing, or personally identifiable data. The method only changes which parameters are updated during post-training and adds no capabilities beyond those of the base models. We are not aware of ethical concerns beyond those already associated with the models and benchmarks used.

REPRODUCIBILITY STATEMENT

The models, training recipe, and baselines are specified in Section 4.1, and the benchmarks and metrics in Section 4.2. Appendix A gives the domain splits, prompts, loss, optimizer settings, expert-subset selection, and evaluation protocol, and Appendix C reports an independent rerun of the main pipeline. We will release all training and evaluation code and the trained modules.

REFERENCES

Tianyu Chen, Shaohan Huang, Yuan Xie, Binxing Jiao, Daxin Jiang, Haoyi Zhou, Jianxin Li, and Furu Wei. Task-specific expert pruning for sparse mixture-of-experts. *arXiv preprint arXiv:2206.00277*, 2022.

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

Damai Dai, Chengqi Deng, Chenggang Zhao, R.X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. 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*, 2024.

Zican Dong, Han Peng, Peiyu Liu, Wayne Xin Zhao, Dong Wu, Feng Xiao, and Zhifeng Wang. Domain-specific pruning of large mixture-of-experts models with few-shot demonstrations. In *Advances in Neural Information Processing Systems*, 2025.

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.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

Yuling Gu, Oyvind Tafjord, Bailey Kuehl, Dany Haddad, Jesse Dodge, and Hannaneh Hajishirzi. OLMES: A standard for language model evaluations. *arXiv preprint arXiv:2406.08446*, 2024.

Suchin Gururangan, Mike Lewis, Ari Holtzman, Noah A. Smith, and Luke Zettlemoyer. DEMix layers: Disentangling domains for modular language modeling. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics*, 2022.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*, 2019.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. LoraHub: Efficient cross-task generalization via dynamic LoRA composition. In *Conference on Language Modeling*, 2024.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *International Conference on Learning Representations*, 2023.

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.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. GShard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*, 2021.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. Branch-train-merge: Embarrassingly parallel training of expert language models. *arXiv preprint arXiv:2208.03306*, 2022.

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*, 2024.

Michael Matena and Colin Raffel. Merging models with Fisher-weighted averaging. In *Advances in Neural Information Processing Systems*, 2022.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, 2022.

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*, 2025.

Jonas Pfeiffer, Sebastian Ruder, Ivan Vulić, and Edoardo Maria Ponti. Modular deep learning. *Transactions on Machine Learning Research*, 2023.

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.

Yikang Shen, Zheyu Zhang, Tianyou Cao, Shawn Tan, Zhenfang Chen, and Chuang Gan. ModuleFormer: Modularity emerges from mixture-of-experts. *arXiv preprint arXiv:2306.04640*, 2023.

Weijia Shi, Akshita Bhagia, Kevin Farhat, Niklas Muennighoff, Pete Walsh, Jacob Morrison, Dustin Schwenk, Shayne Longpre, Jake Poznanski, Allyson Ettinger, et al. FlexOlmo: Open language models for flexible data use. In *Advances in Neural Information Processing Systems*, 2025.

Sainbayar Sukhbaatar, Olga Golovneva, Vasu Sharma, Hu Xu, Xi Victoria Lin, Baptiste Rozière, Jacob Kahn, Daniel Li, Wen-tau Yih, Jason Weston, and Xian Li. Branch-train-MiX: Mixing expert LLMs into a mixture-of-experts LLM. In *Conference on Language Modeling*, 2024.

Ryan Wang, Akshita Bhagia, and Sewon Min. EMO: Pretraining mixture of experts for emergent modularity. Preprint. Model: https://hf.co/allenai/EMO; code: https://github.com/allenai/EMO, 2026a.

Xi Wang, Soufiane Hayou, and Eric Nalisnick. The myth of expert specialization in MoEs: Why routing reflects geometry, not necessarily domain expertise. *arXiv preprint arXiv:2604.09780*, 2026b.

Zihan Wang, Deli Chen, Damai Dai, Runxin Xu, Zhuoshu Li, and Yu Wu. Let the expert stick to his last: Expert-specialized fine-tuning for sparse architectural large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, 2022.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems*, 2023.

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.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super Mario: Absorbing abilities from homologous models as a free lunch. In *International Conference on Machine Learning*, 2024.

Jinghan Zhang, Shiqi Chen, Junteng Liu, and Junxian He. Composing parameter-efficient modules with arithmetic operations. In *Advances in Neural Information Processing Systems*, 2023.

A Experimental details

Domains and splits. Table 5 lists the MMLU subjects in each domain. Following Wang et al. (2026a), each subject's test split is shuffled; the first 60% together with the subject's validation split forms the training set, which also serves as the calibration set for subset selection, and the remaining 40% is the test set. Training and test sets are disjoint; the five dev examples per subject serve only as few-shot demonstrations. GSM8K uses the official train/test splits.

Table 5: Domains, training (= calibration) and test sizes, and optimization steps (3 epochs, batch size 32).

Domain	MMLU subjects	Train	Test	Steps
Math	abstract algebra, college math, elementary math, HS math, HS statistics	752	427	72
Physics	astronomy, college physics, conceptual physics, HS physics	453	257	45
Health	anatomy, clinical knowledge, college medicine, human aging, medical genetics, nutrition, professional medicine, virology	1162	659	111
History	HS European / US / world history, prehistory	658	373	63
Psychology	HS psychology, professional psychology	823	463	78
Law	international law, jurisprudence, professional law	1250	707	120
GSM8K	official splits	7473	1319	702

Prompts and loss. MMLU prompts are 5-shot with the question, four options, and "Answer:"; GSM8K prompts are the 8-shot OLMES chain-of-thought prompts. Training examples are evaluation prompts followed by the gold answer; the loss covers only the tokens after the final "Answer:" plus EOS, truncated to 4096 tokens. Selection, training, and evaluation share the same prompt construction.

Optimization. AdamW (β_1=0.9, β_2=0.999, no weight decay), peak learning rate 10^{-4}, 10% warmup, linear decay, gradient clipping 1.0, bf16, gradient checkpointing, no dropout, 3 epochs,

batch size 32, load-balancing loss weight 0.01. Additional seeds change the data order and initialization only; expert subsets are fixed. The short GSM8K schedule uses learning rate 5×10^{-5} and 1 epoch. LoRA uses rank 16, α=32, and no dropout. Whole-model fine-tuning is sharded across GPUs; learning rate 10^{-4} degrades it (EMO 36.3, standard MoE 35.1 on math), so we report 2×10^{-5}.

Evaluation. MMLU: ranked classification by raw log-likelihood. GSM8K: greedy decoding up to 512 tokens, stopping at "Question:", a blank line, or EOS; the last number is compared by exact match. Context length 2048, bf16.

ESFT. ESFT-Gate and ESFT-Token score each expert on the domain's calibration prompts with the full model's native routing, by its mean gate weight and by the fraction of tokens routed to it, respectively. Experts are added in decreasing score until the cumulative share reaches the threshold p (0.1 for ESFT-Gate and 0.2 for ESFT-Token, the defaults of Wang et al. (2024)). For comparability, ESFT is trained with the same recipe as all other methods.

B Diagnosis: exact values

Table 6 lists the values plotted in Figure 2. Recovered shares are relative to the FP-FT gap of the same run. Retraining the router after math reinsertion recovers 0% of the gap, and interpolating the experts halfway toward their original weights lowers math accuracy from 39.6 to 38.2.

Table 6: Exact values for Figure 2. Top: FP-FT module and its reinsertion (gap: mean±sd over three seeds in parentheses). Middle: share of the FP-FT gap recovered by copying fine-tuned parameters into the reinserted model; attention, norms, and embeddings are copied together with the router rows and the shared expert. Bottom: reinsertion gap when the interface is frozen during training. GSM8K uses the short schedule.

Test	Intervention	MMLU-math	GSM8K
FP-FT	untrained model	35.8	11.5
	trained module, standalone	42.4	21.0
	trained module, reinserted	39.6	12.3
	reinsertion gap	−2.8 (−2.7±1.8)	−8.7 (−9.2±0.4)
H1	+ fine-tuned router rows	18%	2%
H3, restore	+ all interface parameters	68%	71%
	+ attention	66%	79%
	+ norms	60%	30%
	+ embeddings	66%	26%
H3, prevent	reinsertion gap under IF-FT	+1.2 (+0.4±0.7)	−0.5 (−0.6±0.2)

On math, copying attention, norms, or embeddings *without* the router rows and shared expert recovers 40%, 0%, and 7% of the gap. With the full GSM8K schedule, copying attention (with the router rows and shared expert) recovers 74% of the −16.7 gap, and on physics 80% of the −3.9 gap.

C Independent rerun

Table 7 repeats the full IF-FT pipeline, from subset extraction to evaluation, in an independent run. Over three seeds, the reinserted IF-FT models reach 41.22±0.41 on math, 48.79±0.27 on history, and 47.34±0.98 on physics.

D All compositions

Ratios with small denominators are noisy: history and psychology gain only 1.6–2.2 points on their own, so their retention varies widely (0.80–1.83) while the absolute differences stay within one standard error. Giving math priority in the 369 slots it shares with physics yields 41.2 / 44.4 (averaging: 39.1 / 45.1). Composing three modules (math, health, history; 349 shared slots) gives gains of +3.1 / +3.5 / +2.7 against +5.9 / +5.1 / +1.9 alone, $A = 0.72$, and other-domain changes of +2.7 (physics) and +1.5 (psychology). Composition recovers 74%, 64%, and 72% of the pooled

Table 7: Independent rerun of IF-FT: accuracy of the module on its own and after reinsertion (main-run value after the slash), gain over the untrained model, and other-domain change.

Domain	Module	Reinserted (rerun / main)	Reinserted − module	Gain	Other-domain change
Math	40.7	41.0 / 41.7	+0.2	+4.9	+1.4
Physics	48.6	47.5 / 48.2	−1.2	+5.4	+0.4
Health	45.4	47.2 / 47.3	+1.8	+5.0	+1.5
History	47.5	48.8 / 48.8	+1.3	+1.6	+1.1
Psychology	54.6	55.7 / 55.9	+1.1	+2.2	+1.2
Law	31.3	30.6 / 30.8	−0.7	+0.8	+0.7

Table 8: All ten two-module compositions, sorted by subset overlap: gains over the untrained model after composition and for each module alone, retention of each module's gain, pooled additivity, and other-domain change.

Pair (A + B)	ω	Shared slots	Composed gain A / B	Single gain A / B	Retention A / B	A_{ab}	Other-domain change
math + history	0.19	99	+5.7 / +2.7	+5.9 / +1.9	0.97 / 1.42	1.07	+1.8
health + history	0.24	123	+3.9 / +2.9	+5.0 / +1.6	0.79 / 1.83	1.03	+2.0
physics + history	0.24	127	+5.8 / +1.9	+5.4 / +1.6	1.07 / 1.17	1.10	+1.2
history + psychology	0.31	147	+1.3 / +1.7	+1.6 / +2.2	0.83 / 0.80	0.79	+1.5
math + psychology	0.48	233	+3.0 / +1.3	+4.9 / +2.2	0.62 / 0.60	0.61	+1.6
physics + psychology	0.51	253	+3.9 / +2.2	+5.4 / +2.2	0.71 / 1.00	0.80	+0.7
math + health	0.60	299	+3.3 / +3.8	+5.9 / +5.1	0.56 / 0.75	0.64	+2.0
health + psychology	0.65	316	+3.8 / +2.6	+5.0 / +2.2	0.76 / 1.20	0.89	+0.9
health + physics	0.70	345	+3.8 / +4.7	+5.0 / +5.4	0.76 / 0.86	0.82	+0.8
math + physics	0.74	369	+3.3 / +3.1	+5.9 / +6.2	0.56 / 0.50	0.53	+0.8

gain of joint training on the three pairs of Table 3, and 75% for the three-module case; joint training changes psychology by −0.2, +0.2, and 0.0 on those pairs.

E LoRA BASELINES

Table 9: LoRA baselines on EMO: accuracy after training with the shared recipe (learning rate 10^{-4}, 3 epochs, rank 16).

Method	Math	Physics	Health	History
IF-FT	41.7	48.2	47.3	48.8
LoRA, module experts	37.2	42.4	43.6	46.6
LoRA, module attention and experts	37.5	42.8	44.3	47.2
LoRA, full model	37.9	43.2	45.4	48.5

With the shared recipe, the LoRA variants end training at a loss of 1.30–2.03, against 0.38–0.53 for IF-FT, i.e., they underfit at this budget. Combining two full-model LoRA updates by task arithmetic (λ=1) gives 38.9 / 47.7 on math + history and 38.6 / 44.4 on math + physics, below composed IF-FT modules (41.5 / 49.6 and 39.1 / 45.1).

F ESFT DETAILS

With its default thresholds, ESFT selects only 17–43 routed experts in total (0.8–2.1%), because EMO's routing is concentrated: outside the first layer, the top expert of each layer receives 72–93% of tokens, so the cumulative threshold is reached after one to four experts per layer. All of these experts lie inside our expert subsets. Their near-zero other-domain change reflects that few parameters change, not better isolation: after GSM8K training, ESFT-Gate and ESFT-Token lower MMLU-math by 3.3 and 1.9 points, against 0.7 for IF-FT. Capping ESFT's scores at 31 experts per layer yields subsets that overlap ours by 0.84–0.91 and reach the same accuracy, so what limits ESFT on EMO is the number of experts its thresholds admit, not its scoring rule. EMO's subsets are also robust to the selection algorithm: our subset and that of Easy-EP (Dong et al., 2025) share 81% of their experts, with identical zero-shot accuracy.

Table 10: Training inside the full model on EMO: target accuracy and other-domain change. "31/layer": ESFT scores capped at 31 experts per layer. Overlap: fraction of the capped ESFT experts that lie in our expert subset (Gate / Token).

Target	# experts (Gate / Token)	IF-FT w/o extraction	ESFT-Gate	ESFT-Token	Gate, 31/layer	Token, 31/layer	Overlap	Other-domain change (ours / Gate)
Math	21 / 37	41.7	37.2	38.2	42.6	41.5	0.88 / 0.89	+1.7 / +0.2
Physics	21 / 36	49.4	42.0	43.2	50.2	49.4	0.88 / 0.88	+1.1 / +0.2
Health	21 / 40	47.8	43.2	44.9	47.8	48.1	0.88 / 0.89	+1.9 / +0.0
History	17 / 43	49.9	46.1	46.6	50.1	49.9	0.91 / 0.89	+1.6 / +0.2
GSM8K	23 / 40	26.2	19.3	22.2	27.1	26.5	0.84 / 0.85	+0.8 / −0.6

G Training cost

Table 11: Training throughput on H100 GPUs (math, 20 steps, bf16, gradient checkpointing) and parameter counts in billions.

Configuration	GPUs	Per-device batch	Resident	Trainable	Samples/s
IF-FT, module	1	8 / 4 / 1	3.9	3.1	2.59 / 1.80 / 0.45
FP-FT, module	1	8 / 4 / 1	3.9	3.9	2.42 / 1.33 / 0.47
LoRA, module experts	1	8 / 4 / 1	4.0	0.08	1.46 / 0.72 / 0.20
IF-FT without extraction, full model	1	4 / 1	14	3.1	0.48 / 0.13
ESFT-Gate, full model	1	4 / 1	14	0.13	0.43 / 0.12
LoRA, full model	1	4 / 1	14	0.31	0.18 / 0.06
Whole-model fine-tuning	1	1	14	14	0.12
Whole-model fine-tuning, sharded	4	2	14	14	0.77
IF-FT without extraction, sharded	4	2	14	3.1	0.78

Resident parameters are fixed by the method; throughput also depends on the implementation of the MoE layer. In our implementation, which loops over the experts of each layer, the module's 32 experts are processed much faster than the full model's 128, and grouped matrix-multiplication kernels would narrow this gap. Artifact sizes in bf16: IF-FT experts 6.2 GB, whole module 7.8 GB, module LoRA 0.15 GB, full-model LoRA 0.6 GB, full model 27.1 GB.

H Subset overlap

Table 12: Subset overlap ω (top 31 of 127 routed experts, averaged over 16 layers), EMO / standard MoE.

	math	health	physics	history	psych.	GSM8K
math	–	0.60 / 0.64	0.74 / 0.77	0.19 / 0.32	0.48 / 0.56	0.66 / 0.67
health		–	0.70 / 0.72	0.24 / 0.37	0.65 / 0.63	0.46 / 0.49
physics			–	0.24 / 0.32	0.51 / 0.56	0.56 / 0.55
history				–	0.31 / 0.47	0.15 / 0.32
psych.					–	0.36 / 0.40

EMO's subsets for distant domains overlap systematically less than those of the standard MoE (history row: 0.15–0.31 vs. 0.32–0.47).

I A stronger standard MoE

We apply the same protocol on math to Qwen3-30B-A3B (Yang et al., 2025), a standard MoE with 128 experts per layer and no shared expert. Its untrained full model reaches 67.9; the 32-expert module drops to 48.9 after IF-FT training, 19.0 points below the full model, and reinsertion brings it back to 59.3. Reinsertion is thus lossless relative to the module, but the reinserted model remains 8.6 points *below* the untrained full model: on a non-modular MoE, training an extracted subset can hurt the model it came from. Whole-model fine-tuning on the same 752 questions also lowers Qwen3-30B-A3B, to 63.5, suggesting that a small training set limits what any fine-tuning can add to such a strong model. The magnitude is therefore inflated by the strong untrained model, but the sign agrees with the standard MoE in Table 4.

J TASK ARITHMETIC ON THE STANDARD MOE

Fully fine-tuning the standard MoE on math and on history separately (learning rate 2×10^{-5}) and adding the task vectors with coefficient λ gives math / history / psychology of 39.1 / 49.1 / 54.6 (λ=0.3), 41.2 / 49.9 / 54.6 (0.5), 41.2 / 50.1 / 54.0 (0.7), and 37.5 / 47.5 / 51.2 (1.0), against 35.6 / 48.0 / 52.7 untrained. With a tuned λ, task arithmetic on a standard MoE reaches the accuracy of composed EMO modules (41.5 / 49.6), but it requires whole-model fine-tuning of each domain and a coefficient chosen per combination, and at λ=1 both target domains and the other domain fall below the untrained model.