A Brief Technical Summary of Hybrid LoRA

Zhongsheng WANG

School of Computer Science
The University of Auckland

zhongsheng.wang@auckland.ac.nz

October 24, 2025

Paper Mentioned

MIXLORA: Enhancing Large Language Models Fine-Tuning with LoRA-based Mixture of Experts. Dengchun Li et al. Arxiv Preprint, Jun 2024.

Outline

The Challenge of Multi-Task Fine-Tuning

Background. LLMs are deployed across diverse domains (dialogue, reasoning, coding, finance). Adapting one model to many tasks remains non-trivial.

Key Observation.

- Standard fine-tuning $\Rightarrow$ catastrophic forgetting.
- LoRA is efficient but prone to task interference when many adapters coexist.
- MoE improves specialization but adds compute & engineering overhead.

Research Question. *Can we design a parameter-efficient framework that preserves LoRA's simplicity while achieving MoE-level specialization and scalability?*

Related Work – Foundations

PEFT (Parameter-Efficient Fine-Tuning).

- **LoRA** (Hu et al., 2021): freeze W, learn low-rank $\Delta W = BA$; efficient, stable, mergeable.
- **Adapters / AdapterFusion** (2019–2021): task-wise modules; fusion for multi-task.
- **Prefix / Prompt Tuning** (2021–): very light; sometimes weaker on reasoning-heavy tasks.
- **DoRA** (2024): decoupled adaptation for stabilization.

MoE (Mixture-of-Experts).

- **GShard / Switch Transformer** (2020–2021): sparse routing, conditional compute.
- **Mixtral / DeepSeek-V3**: strong specialization at scale; higher system cost.

Takeaway. PEFT ⇒ efficient but limited capacity; MoE ⇒ powerful but costly.

Related Work – Pain Points (Multi-Task)

Task interference.

- Multi-task gradients conflict $\Rightarrow$ mutual drag (learn new, forget old / vice versa).
- One fixed low-rank update struggles to cover heterogeneous skills (vision-language, OCR-like, long reasoning).

Capacity vs. Modularity.

- Per-task LoRA zoo $\Rightarrow$ adapter explosion, conflict on merge, deployment complexity.
- Just raising LoRA rank or orthogonalizing subspaces $\Rightarrow$ still fixed structure, weak dynamic adaptation.

Gap. We need to be efficient, specialized, and dynamically selectable.

Related Work – Attempts & Limitations

A. Bigger MoE.

- More experts $\Rightarrow$ better multi-task, but training/inference/engineering barriers are high on commodity GPUs.

B. Stronger LoRA.

- Orthogonal / subspace-constrained LoRA (e.g., LoRI) mitigates conflicts but remains fixed low-rank, lacking input-dependent flexibility.

C. Naive LoRA+MoE.

- Stacking multiple LoRAs as "experts" without factor-level coordination (between A/B) or tight routing–update coupling $\Rightarrow$ limited gains.

Conclusion. Existing paths are either strong but expensive or cheap but rigid. The dynamic low-rank expert gap remains.

Motivation: Why It Matters

LLM fine-tuning in the real world.

- **Industry:** domain copilots (finance/healthcare/legal), retrieval-augmented assistants.
- **Creative AI:** multi-character storytelling, dialogue agents, tutoring systems.
- **Research:** multi-domain benchmarks (MMLU, BigBench) require balanced accuracy vs. efficiency.

Pain point.

- Dozens of LoRAs across domains $\Rightarrow$ parameter bloat, interference, deployment complexity.
- Pure MoE is costly at training & inference, hard to engineer on a single GPU.

Goal. Scalable, modular, efficient adaptation — without retraining from scratch or losing specialization.

From the Dilemma to MixLoRA

Motivating Question. Can we combine LoRA's low-cost efficiency with MoE's high-capacity specialization?

Observation.

$$\text{LoRA (dense)} \Rightarrow \text{cheap but narrow capacity}$$

$$\text{MoE (sparse)} \Rightarrow \text{powerful but expensive}$$

Our Insight. Embed LoRA within MoE — let each expert carry a lightweight LoRA adapter, and use a router to select a sparse subset per token.

This leads to → MixLoRA: Efficient, specialized, dynamically routed low-rank experts.

Preliminary: LoRA – Low-Rank Adaptation

Core Idea. Instead of updating all parameters, LoRA freezes the pre-trained weights W and learns a low-rank update:

$$W' = W + BA,$$

where $A \in \mathbb{R}^{r \times d_2}$ and $B \in \mathbb{R}^{d_1 \times r}$ are trainable, and $r \ll \min(d_1, d_2)$.

Mathematical Intuition. Fine-tuning updates ΔW in LLMs tend to be low-rank:

$$\Delta W \approx BA, \quad \text{so we only train } A, B.$$

Key Points.

- Parameter efficiency: train $O(r(d_1 + d_2))$ instead of $O(d_1 d_2)$.
- Stable adaptation: the pretrained model remains frozen.
- Linear composition: $W' = W + \sum_k B_k A_k$ enables multi-LoRA merging.

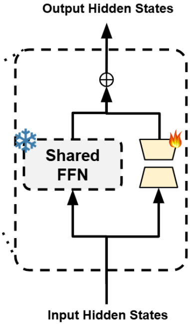
Cont.

How it works.

- LoRA inserts two small trainable matrices (A, B) into the frozen network.
- During forward pass, the pre-trained weight W stays fixed – only the low-rank path learns.
- At inference, the low-rank update BA is merged into W.

Why it helps.

- Captures task-specific variation without disturbing base knowledge.
- Easy to swap: one backbone, many small adapters.
- Reduces GPU memory and fine-tuning time drastically.



LoRA injects low-rank adapters into FFN or attention layers.

Preliminary: Mixture-of-Experts (MoE)

Core Idea. MoE replaces a single feed-forward network (FFN) with multiple experts. For each token, only a few experts are activated:

$$h' = \sum_{i=1}^{N} R(h)_i \, E_i(h),$$

where E_i is the i-th expert and $R(h)$ is a routing weight vector.

Routing.

$$R(h) = \text{Top-K}\big((W_r h + \text{noise})\big),$$

where W_r is trainable and $K \ll N$ controls sparsity.

Key Insights.

- **Conditional computation:** Only K experts are active per token.
- **High capacity:** Adds parameters without raising FLOPs.
- **Specialization:** Experts focus on different skills or domains.

Cont.

How it works.

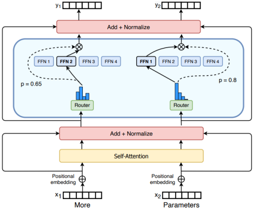
- Each token passes through a **router** that scores all experts.
- Router selects **Top**-K experts (usually $K=1$ or 2).
- Selected experts process the token; outputs are weighted and merged.

Why it helps.

- Increases model capacity without higher FLOPs.
- Encourages expert specialization across domains.
- Supports large-scale training efficiently.

Challenges.

- **Imbalance:** Some experts overused, others idle.
- **Overhead:** Token dispatch and aggregation add cost.



MoE layer: tokens $\to$ router $\to$ Top-K experts $\to$ aggregation.

(Adapted from Switch Transformer)

Method: MixLoRA Architecture

Key Idea.

- Embed LoRA adapters inside an MoE framework.
- Each expert is a LoRA pair $(A_k^{(\ell)}, B_k^{(\ell)})$ on shared $W^{(\ell)}$.
- Router dynamically activates a sparse subset of experts per input token.

Formulation

$$E_k^{(\ell)}(h) = W^{(\ell)}h + B_k^{(\ell)}A_k^{(\ell)}h, \qquad h^{(\ell)} = \sum_{k=1}^{K} R^{(\ell)}(h)_k \, E_k^{(\ell)}(h).$$

Interpretation. Each expert contributes a low-rank, task-specific update on a shared FFN backbone.

MixLoRA Architecture Visualization

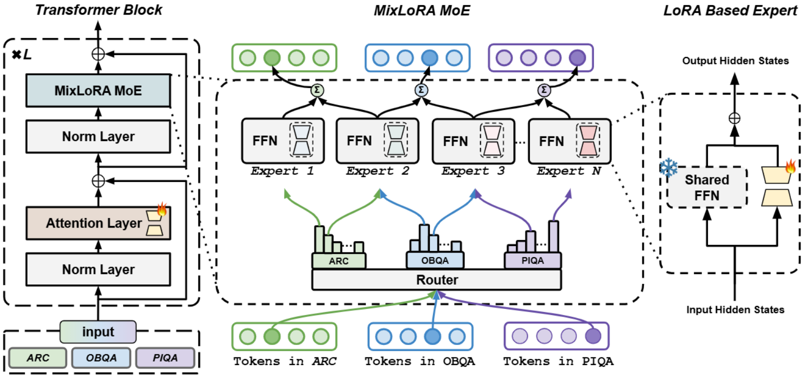


Illustration: shared FFN backbone with per-expert LoRA adapters and a Top-K router selecting active experts.

Routing and Expert Integration

Routing Mechanism.

- The router assigns scores $R^{(\ell)}(h)$ to each expert.
- Only Top-K experts are activated for each token (sparse computation).
- Load-balancing loss encourages uniform expert usage.

Formulation

$$R^{(\ell)}(h) = \text{Top-K}\big((W_r^{(\ell)} h + \text{noise})\big), \quad K \ll N.$$

$$h^{(\ell)} = \sum_{k=1}^{K} R^{(\ell)}(h)_k \, E_k^{(\ell)}(h).$$

Training.

- Jointly train the router and LoRA adapters while keeping the backbone frozen.
- Use auxiliary loss to prevent expert collapse.
- Sparse gradient flow: only the Top K experts backpropagate.

Routing and Expert Integration (Visualization)

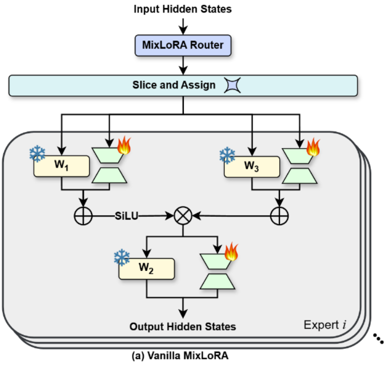


Illustration: tokens are scored by the router, Top-K experts are selected, and outputs aggregated.

Experimental Setup

Base Models.

- Experiments are conducted on **GPT-style** and **LLaMA-style** transformer backbones.
- Each FFN layer is replaced by a **MixLoRA block** with $K = 2$ experts.
- The backbone weights $W^{(\ell)}$ are frozen; only LoRA adapters and routing parameters are trainable.

Training Configuration.

- Optimizer: AdamW with learning rate 1×10^{-4}, weight decay 0.01.
- Batch size: 64–128, sequence length 512.
- Training for 3–5 epochs depending on dataset size.
- **Auxiliary balancing loss** added to the main objective:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda \mathcal{L}_{\text{balance}}, \quad \lambda = 0.01.$$

Cont.

Tasks and Benchmarks.

- Multi-domain instruction tuning covering *reasoning, dialogue, commonsense, and code*.
- Benchmarks include ARC-e/c, BoolQ, PIQA, SIQA, HellaSwag, WinoGrande, etc.
- Metrics: Accuracy / F1 for classification and QA; Rouge / BLEU for generation.

Baselines.

- **LoRA Series:** dense low-rank adapters on all layers (DoRA, etc.).
- **MoE:** traditional Top-K expert routing without LoRA compression.
- **MixLoRA (ours):** shared FFN + per-expert LoRA + sparse routing.

Main Results on Multi-Task Benchmarks

Goal. Evaluate MixLoRA against LoRA and MoE on multi-domain reasoning benchmarks, focusing on both accuracy and parameter efficiency.

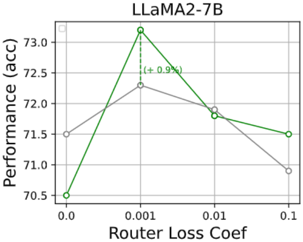
Metrics. Each cell shows task accuracy (%). "AVG." is the average over all benchmarks.

Model	#Params (M)	ARC-e	ARC-c	BoolQ	OBQA	PIQA	SIQA	HellaS	WinoG	AVG.
LoRA	180	76.3	55.2	83.1	68.4	79.7	48.3	79.8	73.4	70.5
MoE	200	77.0	56.1	83.5	69.0	80.3	49.2	80.4	74.0	71.2
MixLoRA (ours)	**190**	**78.8**	**57.8**	**84.3**	**70.4**	**81.2**	**50.1**	**81.0**	**75.2**	**72.4**

Key Observations.

- MixLoRA achieves the **highest average accuracy (+2.0–2.5%)** while using fewer parameters than MoE.
- Gains are consistent across all domains (scientific, physical, and social reasoning).
- Low-rank expert design improves multi-task generalization and avoids overfitting.

Ablation: Router Loss Coefficient

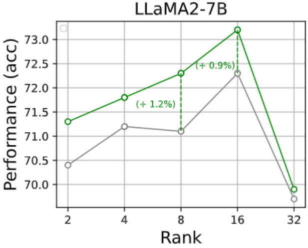


Effect of router-loss coefficient on average accuracy (ARC, OBQA, BoolQ).

Findings.

- Optimal coefficient $\lambda = 10^{-3}$ yields best average accuracy.
- Too large λ impedes convergence; disabling loss causes imbalance.
- Auxiliary loss effectively balances expert workloads.

Ablation: LoRA Rank & Efficiency

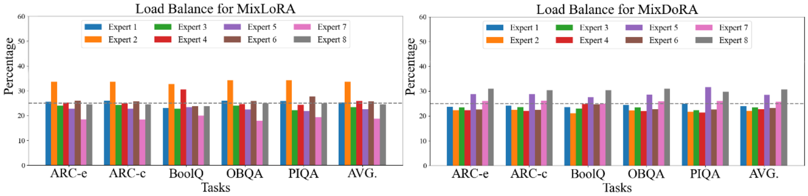


Impact of LoRA rank on performance and computational efficiency.

Observations.

- Stable performance for ranks $r = 2$–16; degraded at $r = 32$.
- MixLoRA maintains 30% lower token latency vs. vanilla MoE.
- Balance loss reduces expert collapse and ensures uniform activation.

Ablation: Expert Load Distribution



Distribution of token assignments across experts during training.

Observations.

- MixLoRA achieves nearly uniform expert utilization – standard deviation of load ≈ 0.0223.
- Without auxiliary loss, a few experts dominate (expert collapse).
- Balanced routing improves robustness and multi-domain generalization.

Conclusion. Auxiliary balance loss ensures fair expert activation, stabilizing training and improving overall performance.

Takeaways

- MixLoRA fuses **LoRA's parameter efficiency** with **MoE's expert specialization**.
- Sparse routing + load balancing yield strong multi-task performance at low cost.
- A practical recipe for resource-constrained adaptation of LLMs.

Open Question: Beyond MixLoRA

Observation. MixLoRA alleviates task-level interference via expert routing. However, similar representation conflicts also appear in other domains.

Example. In text-to-image, composing multiple LoRAs (identities, styles) often causes feature entanglement/identity collision.

Open Question.

- How to prevent/disentangle conflicts when multiple LoRAs are composed on the same backbone?
- Can routing/modular principles inspire non-conflicting multi-identity/multimodal generation?

Takeaway. Task-level interference is mitigated; cross-LoRA conflict remains open.

Case Study

- Train different LoRA extensions for different characters.
- Two different people superimposed (LoRA superimposed) create conflicts in the generated image.

References

1. MIXLORA: Enhancing Large Language Models Fine-Tuning with LoRA-based Mixture of Experts. Dengchun Li et al. Arxiv Preprint.
2. LoRA: Low-Rank Adaptation of Large Language Models. Edward Hu et al. Accepted by ICLR 2022 (Poster).
3. A Comprehensive Survey of Mixture-of-Experts: Algorithms, Theory, and Applications. Siyuan Mu et al. Arxiv Preprint.
4. CharCom: Composable Identity Control for Multi-Character Story Illustration. Zhongsheng Wang et al. ACM MMAsia 2025.
5. DoRA: Weight-Decomposed Low-Rank Adaptation. Shih-Yang Liu et al. ICML 2024 Oral.
6. GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding. Dmitry Lepikhin et al. ICLR 2021 Poster.
7. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. William Fedus et al. JMLR 2022.

Thanks for listening!

Zhongsheng WANG

October 24, 2025