LiME

Lightweight Mixture of Experts for Efficient Multimodal Multi-task Learning

Md Kowsher · Haris Mansoor · Nusrat Jahan Prottasha · Ozlem Garibay
Victor Zhu · Zhengping Ji · Chen Chen

University of Central Florida · Coventry University · Axon

4× fewer parameters | **29% faster training** | **zero-parameter routing** | **any PEFT method**

THE PROBLEM

Combining MoE with PEFT is expensive

Recent MoE-PEFT methods add a separate adapter for every expert. Trainable parameters then grow linearly with the number of experts — fighting the very goal of parameter-efficient fine-tuning.

Parameter explosion

Adapters are replicated per expert — cost scales as $E \times |\phi|$.

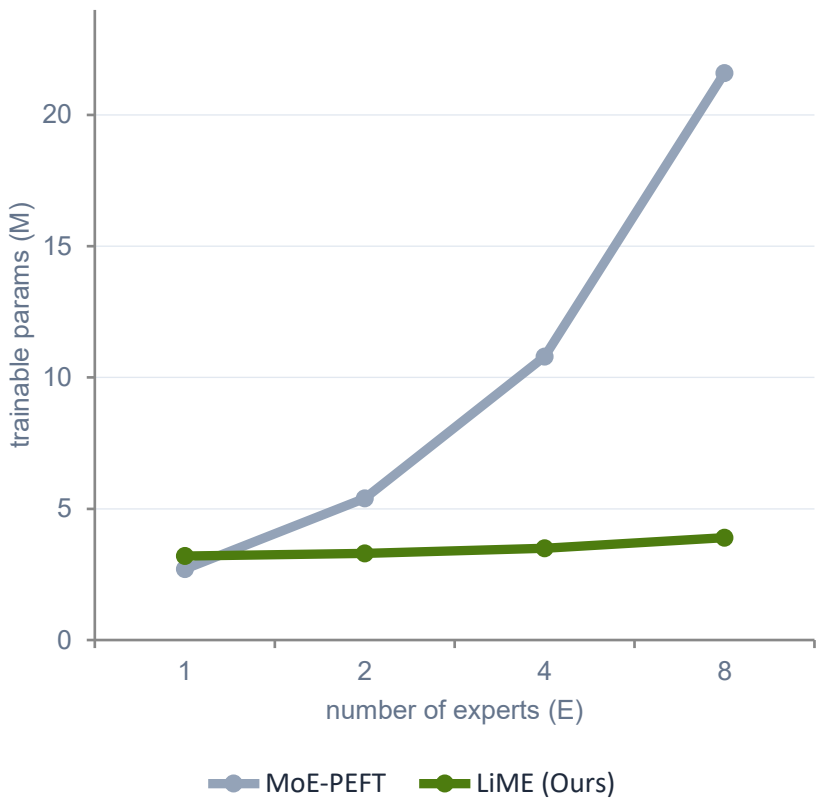
Router overhead

A learned router adds $d \times E$ parameters in every layer.

Architecture lock-in

Designs are tied to LoRA-style adapters, excluding other PEFT.

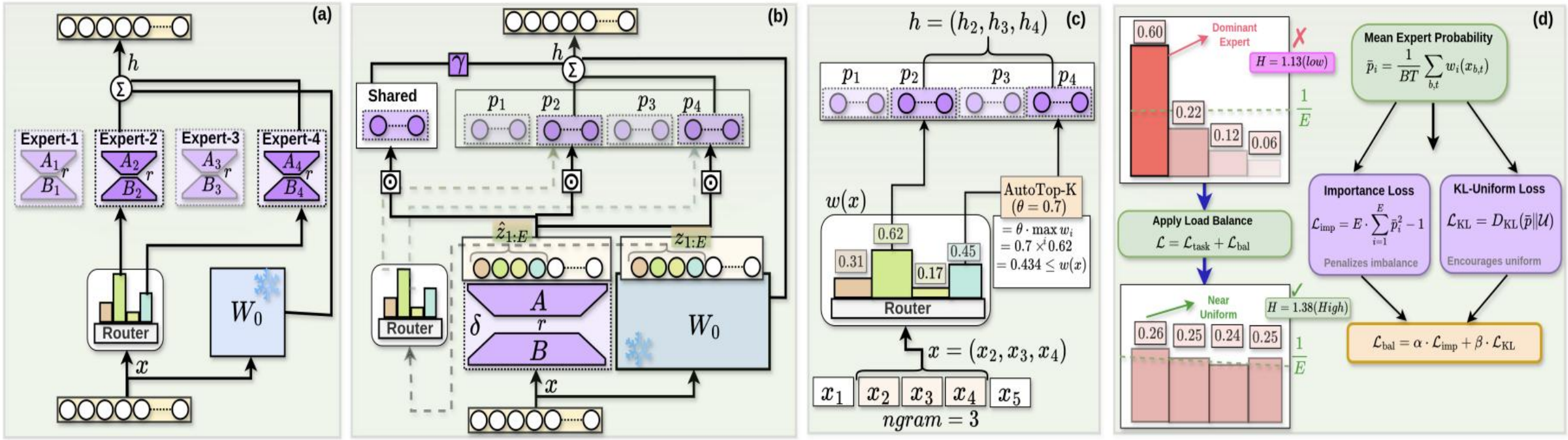
Trainable params grow with experts



OUR IDEA

Specialize by modulation, not replication

LiME keeps one shared PEFT module and gives each expert only a lightweight scaling vector — expert behavior emerges from rescaling, not from duplicating adapters.



METHOD

Two components, both nearly free

Lightweight experts

Rescale a shared PEFT output element-wise with routed expert vectors.

$$h = z + \hat{z} \odot P(x), \quad P(x) = \Sigma_i \, w_i(x) \cdot p_i$$

Params per layer: $|\phi| + E \cdot d_o$ vs. $E \times |\phi|$

Works with any PEFT: LoRA, DoRA, LoRA-FA, SliceFine, Prompt Tuning.

Zero-parameter routing

Route from representations already computed in the forward pass — no learned router at all.

$$w(x) = \mathrm{softmax}([(1-\gamma_r) \cdot \hat{z}_{1:E} + \gamma_r \cdot \hat{\tilde{z}}_{1:E}] / \tau)$$

frozen output z + PEFT output ẑ

A tiny E-dimensional slice ($E \ll d$) carries enough signal to choose among E experts.

PRACTICAL MECHANISMS

Making the mixture train well

Auto Top-K

Adaptive expert selection by confidence:

$$w_i \geq \theta \cdot \max_j w_j$$

Few experts when routing is sharp, more when it's uncertain (avg ≈ 1.7).

N-gram routing

One routing decision per window of n tokens.

$$\text{last-token rep (causal)}$$

Encourages locally coherent expert assignments; n = 3 by default.

Load balancing

Importance + KL-uniform auxiliary losses.

$$\mathcal{L} = \mathcal{L}\text{task} + \alpha\mathcal{L}\text{imp} + \beta\mathcal{L}\text{KL}$$

Prevents expert collapse while preserving useful specialization.

THEORY

Three guarantees behind the design

Theorem 1 — more experts preserve information

Adding experts refines the input partition and cannot lose task-relevant information.

$$I(Y; Z_n) \geq I(Y; Z_{n-1})$$

Theorem 2 — modulation ≈ expert-specific PEFT

Shared PEFT with modulators approximates separate adapters within a bounded risk gap.

$$| R^*(Z\_LiME) - R^*(Z\_MoE) | \leq O(\bar{e})$$

Theorem 3 — last token is the best router

Under causal attention the final position in each window is the most informative for routing.

$$I(Y; h_n) \geq \cdots \geq I(Y; h_1)$$

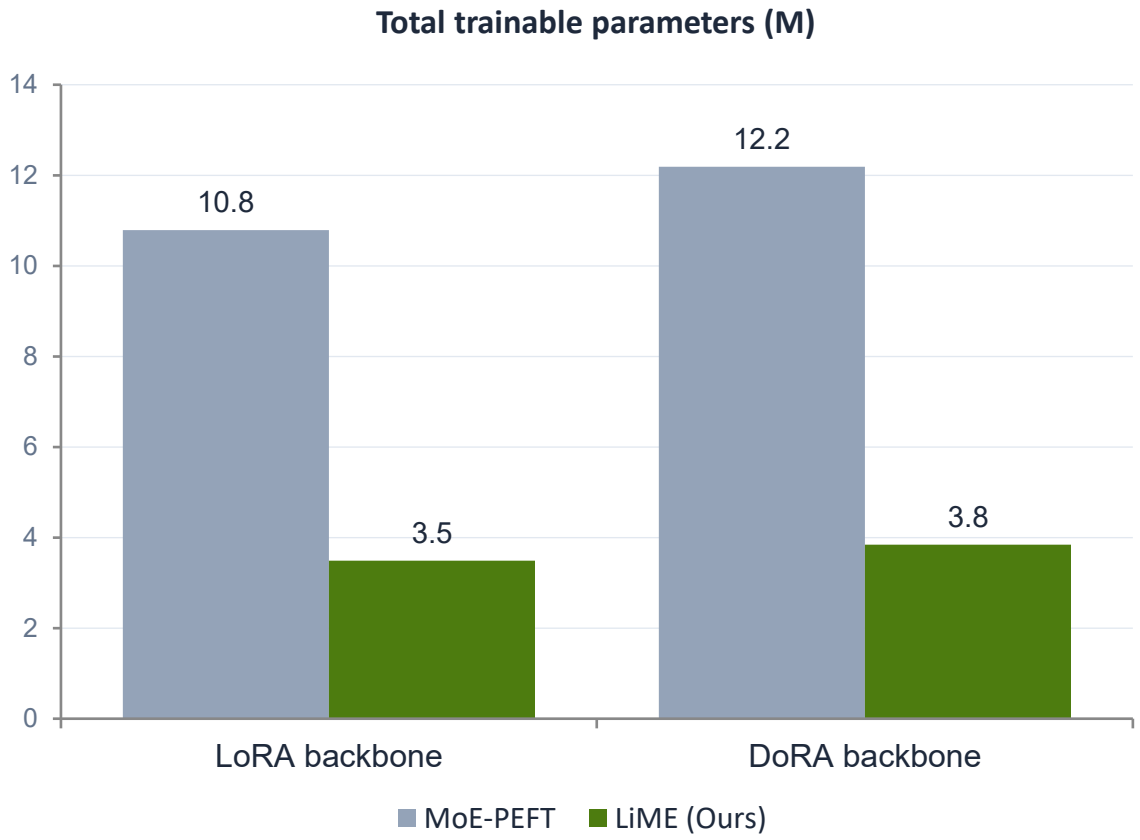
RESULTS · EFFICIENCY

Far lighter, noticeably faster

4× fewer trainable parameters than matched MoE-PEFT

29% faster training (LiMEDoRA: 35.5 vs 50.2 min)

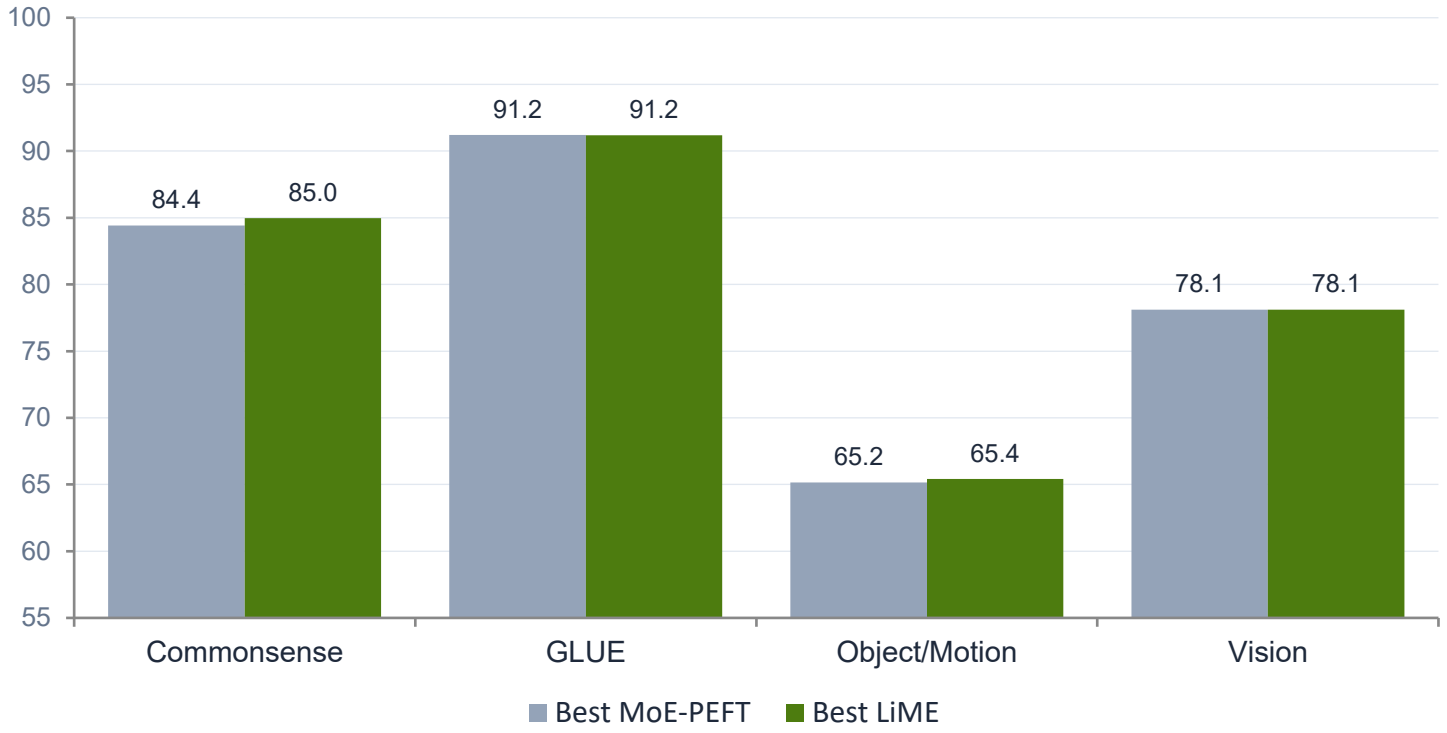
0 router parameters added per layer



RESULTS · PERFORMANCE

Competitive or better across 47 tasks

On MMT-47 (text, image, video; 158K training samples), LiME matches or beats strong MoE-PEFT baselines while training a fraction of the parameters.



What stands out

- Beats its own base PEFT on every category (+1.6–2.1% avg).
- Stable as experts scale; MoELoRA drops 8–10% beyond E = 4.
- Mean CKA 0.935 with full MoE-PEFT representations.

Takeaways

Expert specialization can come from modulation, not adapter *replication*.

Lightweight experts

Rescale a shared PEFT output — works with any PEFT method.

Zero-parameter routing

Reuse frozen + adapted features; no learned router.

Practical & principled

Auto Top-K, n-gram routing, load balancing — backed by theory.

LiME — competitive accuracy, up to 4× fewer parameters, 29% faster. Thank you!

Code & dataset (MMT-47) released · Correspondence: ga.kowsher@gmail.com