Latent-LoRA: Compact Latent-Space Adapters with Gradient-Free Routing for Continual Learning

Literature Review

Hiteshwari Patel

NIT SURAT

Sunday, 1 November 2026

Idea 1: Replay-Free Continual Learning

Where This Fits

Our Pilot

O-LoRA
Plain LoRA

Both finished below the frozen base model.

Harsh's Focus

O-LoRA
OPLoRA

Protecting previously acquired knowledge.

Today's Paper

Latent-LoRA

Choosing adapters without a trained router.

Central Distinction

Protecting knowledge is one problem. Selecting the right knowledge at test time is another.

The Story So Far



The Key Question

Once we have multiple adapters, which ones should be activated for a new input?

The Problem

Objective

Learn tasks sequentially without:

- Replaying previous task data.
- Providing task identity at inference time.

O-LoRA

- Encourages new adapters to be orthogonal to old ones.
- Adds all adapters during inference.
- Every adapter can affect every input.

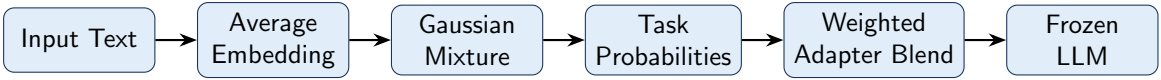
GainLoRA

- Learns a gate for each task.
- Gates can themselves be forgotten.
- Parameters grow with every task.

Open Question

Can we select adapters cheaply and reliably without training an additional router?

The Idea in One Picture



Tiny Adapters

Built-in Orthogonality

Training-Free Router

1. Compact Latent-Space Adapters

Step 1: Apply Singular Value Decomposition to the original weight matrix.

$$W = U\Sigma V^T$$

The decomposition is computed once and frozen.

Step 2: Each task learns only a small matrix:

$$R_t \in \mathbb{R}^{r \times r}$$

The task-specific update is:

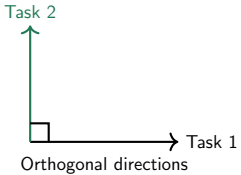
$$\boxed{\Delta W_t = U\Sigma R_t V^T}$$

Parameter Efficiency: Llama-3-8B

4.2M	**66K**
Standard LoRA	Latent-LoRA

Approximately 64× fewer trainable parameters per task

2. Right Angles with a Guarantee



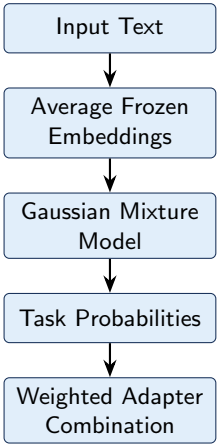
- A penalty encourages orthogonality between task matrices.
- The paper theoretically bounds cross-task interference for any input.
- Standard LoRA penalties do not provide the same guarantee.

Ablation: SuperNI, Order 1, No-Router Setting

$33.30 \longrightarrow 16.71$

Average performance without the orthogonality penalty.

3. A Router with No Gradient-Based Training



- Represent input by average frozen token embeddings.
- Model each task via GMM (5 components, shared covariance).
- Task probabilities define weighted adapter blend.
- Adding a task takes $\sim$2–5 s on CPU.

Important

No router gates are trained; relies strictly on precomputed training data statistics.

How Did They Measure It?

Benchmarks

- SuperNI & Long Sequence (15 tasks each)
- Evaluated across two task orders

Models

- T5-Large, T5-XL
- Llama-2-7B, Llama-3-8B, Llama-2-13B

Baselines

- O-LoRA, InfLoRA, GainLoRA

Protocol

- 3 random seeds; multiple task orders
- Metrics: AP (↑) and FM (↓)

Evaluation Metrics

- **Average Performance (AP ↑):** Mean score across tasks after the final task.
- **Forgetting Measure (FM ↓):** Mean performance drop from peak observed scores.

Results: Latent-LoRA vs Baselines

T5-Large | SuperNI (Order 1)

Method	AP ↑	FM ↓
lightgreen **Latent-LoRA**	**48.60**	**0.01**
GainLoRA	46.77	2.14
InfLoRA	39.02	8.02
O-LoRA	27.91	17.23

Llama-3-8B | SuperNI

Latent-LoRA: AP ≈ 56, FM ≈ 0.01 | **O-LoRA**: AP ≈ 43, FM $\in [6, 9]$.

Router Performance

Near-perfect task posteriors on SuperNI; consistently high on Long Sequence.

Reported by paper authors; not independently reproduced by our team.

Limitations and What We Can Use

Limitations

- Assumes tasks are separable in average embedding space.
- Shared covariance inversion has $O(d^3)$ computational cost.
- No evaluation above 13B parameters.

What We Can Use

- Frozen-embedding router.
- Compact task-specific adapters.
- AP and FM evaluation protocol.
- Multiple task orders and seeds.

Things to Watch for in Our Setting

- Paper uses soft blending; we planned hard routing.
- Legal subdomains may be much closer than SuperNI tasks.
- Stored mixture statistics may conflict with no-retention requirements.

Questions for the Group

1. Can frozen-embedding routing separate closely related legal subdomains?
2. Should we use soft blending or hard routing when mistakes are costly?
3. Do stored Gaussian mixture statistics violate a no-retention requirement?
4. Can an $r \times r$ adapter learn enough from only 40–60 training examples?



Next Step: Test Task Separability