Orthogonal Subspace Learning for Language Model Continual Learning

O-LoRA (Wang et al., Fudan University, Findings of EMNLP 2023)

Harsh Raj - IIT Mandi - GroupBrain Talks

25 October 2026

Outline

1. Introduction
2. Background: LoRA
3. The O-LoRA Method
4. Results & Limitations
5. Connection to our Research
6. Summary
7. References

The Problem: Catastrophic Forgetting

- Real systems meet tasks **one after another**, not all at once.
- Fine-tuning on a new task **overwrites** earlier knowledge: **catastrophic forgetting**.
- Goal: a model that
  - **keeps** old tasks while **learning** new ones,
  - and still **generalizes** to unseen tasks.

Setting

Tasks $\mathcal{D}_1, \ldots, \mathcal{D}_T$ arrive as a stream. At step t only $\mathcal{D}_t$ is visible: **no replay, no historical data**. At test time the **task ID is unknown**.

Existing Approaches and Their Limits

Rehearsal

Replay stored examples with the new task.

Limit: privacy risk, growing cost.

Regularization

Penalize changes to "important" weights.

Limit: struggles on long task sequences.

Architecture

Add or isolate parameters per task.

Limit: per-task experts, weak transfer to unseen tasks.

The Gap

Orthogonal gradient descent updates **orthogonally to old-task gradients**, but needs to **store those gradients**: infeasible for LLMs. O-LoRA finds a cheap proxy for them.

Background: LoRA

- Weight updates of pre-trained models have a **low intrinsic dimension**.
- Freeze $W_{\text{init}} \in \mathbb{R}^{d \times k}$ and learn only a low-rank update:

$$h = W_{\text{init}} x + ABx, \qquad A \in \mathbb{R}^{d \times r},\ B \in \mathbb{R}^{r \times k},\ r \ll \min(d, k)$$

Why it matters here

The columns of A span the directions in which the layer output is changed. O-LoRA constrains exactly these directions, on attention W_q and W_v.

Key Insight: LoRA Subspace as a Gradient Proxy

Hypothesis

The LoRA parameters of a task **succinctly capture its gradient subspace**.

- Old-task gradients are not stored: the **LoRA weights are the proxy**.
- Learn each new task in a subspace **orthogonal** to the old LoRA subspaces.
- The part of an update lying inside the old subspace is what causes forgetting.

O-LoRA: One Low-Rank Subspace per Task

- Task t gets its own LoRA $\{A_t, B_t\}$. Its update subspace is spanned by the columns of A_t:

$$\mathcal{U}_t = \text{span}\{a_t^1, \ldots, a_t^r\}$$

- Subspaces are orthogonal when every cross inner product vanishes, which reduces to one matrix condition:

$$\boxed{O_{i,t} = A_i^\top A_t = 0 \qquad (i < t)}$$

Training Objective

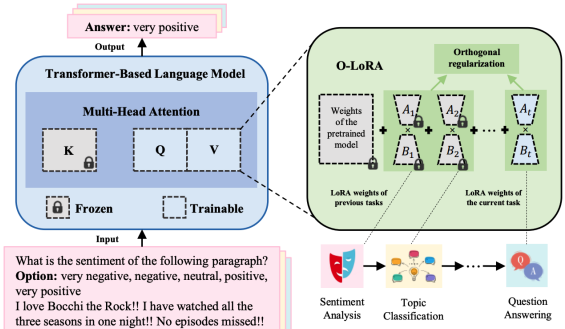
$$\mathcal{L}_t = -\sum_{(x,y)\in\mathcal{D}_t} \log p_\Theta(y \mid x) \;+\; \lambda_1 \sum_{i=1}^{t-1} \mathcal{L}_{\text{orth}}(A_i, A_t), \qquad \mathcal{L}_{\text{orth}} = \|A_i^\top A_t\|_F^2$$

1. **Freeze** the base model and all previous LoRA $\{A_i, B_i\}_{i<t}$.
2. **Train** only A_t, B_t with the loss above; λ_1 sets the penalty strength.
3. **Apply** LoRA on the attention **query** (W_q) and **value** (W_v) weights.

Cost

The penalty is a tiny $r \times r$ product per layer, and no old data is needed.

O-LoRA at a Glance



Source: Wang et al. (2023), Figure 2

- Old LoRA branches **frozen**; current branch **trainable**.
- Orthogonal regularization keeps $A_t \perp A_i$.
- One **instruction format** for all tasks (definition, options, input): **no task ID needed at test time**.
- After training, **merge**: $W_{\text{init}} \leftarrow W_{\text{init}} + \sum_i A_i B_i$, so no memory growth.

O-LoRA: Achievements and Results

Standard CL benchmark

T5-large, average accuracy after the last task

O-LoRA 75.8
LFPT5 72.7 (+3.1)
Per-task FT 70.0
Multitask 80.0 (upper bound)

Long sequence (15 tasks)

Same metric

O-LoRA 69.6, on par with LFPT5 69.2
ProgPrompt 77.9, but it **needs task IDs**

Unseen tasks (zero-shot MMLU)

LLaMA-7B after the CL sequence

O-LoRA 33.6
Plain LoRA baselines 23.3 / 28.6 (near random)
Before CL: 37.5

Takeaway

Best replay-free result on the standard benchmark and the only method that is replay-free, parameter-efficient, task-ID-free and keeps zero-shot ability. Forgetting is **reduced, not eliminated**.

Limitations

Stated by the authors

- Hundreds of tasks are untested.
- Task boundaries are still needed *during training*, to allocate one LoRA per task.
- Long sequences remain hard: continual-learning methods stay below per-task and multitask training.

How O-LoRA Helps our Research

What we can take from it

- Same constraint as my setting: a small local LLM learns new legal material with **no replay of confidential documents**.
- A principled **baseline**: one adapter per task, no stored data.
- A clear **failure mode** to study: adapters from all tasks are summed at inference.

What the early pilot test found

On an 8B 4-bit model, every adapted variant (sequential LoRA, cumulative adapters, O-LoRA) finished **below the frozen base** (57.81–60.94% vs 65.62%), and the orthogonality penalty **changed none of the final predictions**.

My extension: isolate and route

Give each legal domain its own adapter and **route each query to a single one** with a training-free embedding router. Measure **acquisition, retention, routing accuracy and misrouting loss**, especially between close subdomains.

References & Further Reading

Paper:

- X. Wang et al., *Orthogonal Subspace Learning for Language Model Continual Learning*, Findings of EMNLP 2023 (arXiv:2310.14152).
- Code: `https://github.com/cmnfriend/O-LoRA`

Building blocks:

- Hu et al., *LoRA: Low-Rank Adaptation of Large Language Models*, 2021
- Farajtabar et al., *Orthogonal Gradient Descent for Continual Learning*, 2020

Thank you — questions?