HydraLoRA: An Asymmetric LoRA Architecture for Efficient Fine-Tuning

Chunlin Tian Zhan Shi Zhijiang Guo Li Li Chengzhong Xu

University of Macau

NeurIPS 2024 Oral

Presenter: Haotian Liu

November 11, 2024

Outline

Outline

Motivation

With the development of LLM, people tend to adapt a single LLM for multiple downstream applications via fine-tuning to cater to specific domain needs.

FFT(Full fine-tuning)

- All the parameters will be updated during the training process.
- Cost extensive memory and computational resources.

PEFT(Parameter-Efficient Fine-tuning)

- Freeze the backbone model parameters while only a minimal number of task-specific parameters are introduced and fine-tuned.
- Lower parameters lead to higher efficiency, but compromised quality in target domains characterized by complex sub-domains and diverse tasks.

Motivation

Compelling research question:

- What is the optimal architecture that can deliver superior model performance and exhibit robust generalization across unseen tasks while still capitalizing on the efficiency benefits of a reduced parameter footprint?

Background

LoRA Basics:

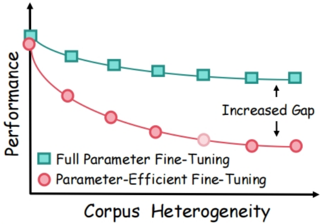
- Freeze pre-trained model weights W_0 and insert trainable rank decomposition matrices into a layer of the pre-trained model.

$$y' = y + \Delta y = W_0 x + BAx$$

where $y \in R^d$ is the output and the $x \in R^k$ denotes the input. $B \in R^{d\times r}$, $A \in R^{r\times k}$ with $r \ll min(d, k)$.

LoRA's Practical Dilemma

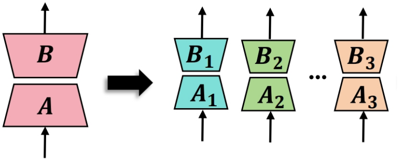
- While restricting the number of tuned parameters is essential for training efficiency, it hinders the model's ability to learn from diverse datasets.



Background

Observation I:

- Leveraging data from diverse tasks within a domain, and training distinct LoRA heads for each task. Tuning with Dolly-15K and evaluating on MMLU.



(a) (b)

Schemes	$r \times n$	MMLU ↑	% Parameter
LoRA	8×1	43.22	0.062
LoRA	16×1	45.45	0.124
LoRA	32×1	46.59	**0.248**
LoRA (Split)	16×2	46.82	0.248
LoRA (Split)	8×4	**46.94**	0.248
LoRA (Split)	4×8	46.83	0.248

- With the same parameter count, rather than employing a single LoRA for the entire domain dataset, it is more effective to deploy multiple smaller LoRA heads, each dedicated to a specific downstream task.

.

Background

Observation II:

- Employ the t-SNE technique to visualize the parameters of matrix A and B across all heads.

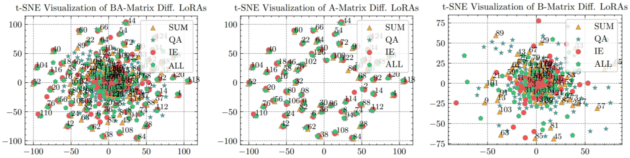


Figure 3: Breakdown analysis of LoRA modules. Compare fine-tuned LoRA modules of Dolly-15K [8] with three subtasks of Dolly-15K including "*summarization (Sum)*", "*closed QA (QA)*" and "*information extraction (IE)*" using t-SNE. Consider LLaMA2-7B (random seed=42), which contains 32 decoder layers, corresponding to 32 adaptive modules. Each module consists of {**0**: q_proj of A, **1**: q_proj of B, **2**: v_proj of A, **3**: v_proj of B} submodules. This makes a total of 32×4 submodules. Left displays all submodules. Center shows all even submodules, i.e. the A matrix. Right represents all odd submodules, i.e. the B matrix. It can be seen that the differences in the fine-tuned LoRA modules for different tasks arise mainly from the B matrix.

- When multiple LoRA heads are trained individually on different data, the parameters of matrix A from different heads tend to converge, while those of matrix B are distinguishable.

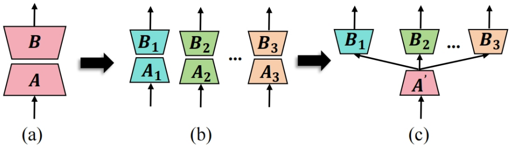
Outline

Asymmetric LoRA architecture

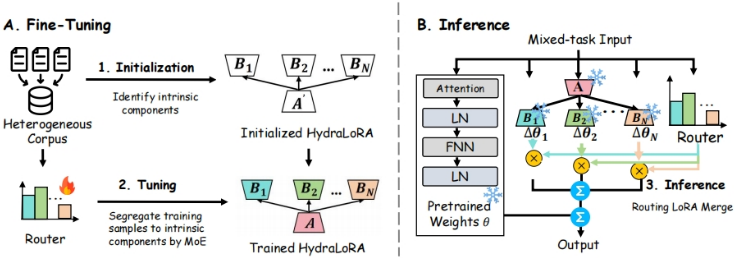
A central shared matrix A and several distinct matrices B.

$$
\begin{aligned}
W &= W_0 + \Delta W \\
&= W_0 + \sum_{i=1}^{N} \omega_i \cdot B_i A
\end{aligned}
$$

The matrics $B_i \in \mathcal{R}^{d \times r}$ and shared $A_i \in \mathcal{R}^{r \times k}$. The hyper-parameter N denotes the number of B matrices. The term ω_i modulates these contribution weights for head B_i.



Workflow of HydraLoRA



1. Specify the number of tasks by applying k-means or developer-specified size.
2. During finetuning, use the Mixture-of-Experts (MoE) framework to handle B matrices as expert adapters.
3. During inference, flexibly and dynamically merges multiple B matrices through the MoE router.

Initialization via k-means

- First, extract key features from the corpus by applying the Term Frequency-Inverse Document Frequency (TF-IDF) algorithm and transform the textual information into numerical feature vectors.
- Then, integrate the elbow method to determine the optimal value of K.

$$C_j = \arg\min_{C_j} d(X_i, C_j) \tag{1}$$

$$C_j = \frac{1}{|S_j|} \sum_{X_i \in S_j} X_i \tag{2}$$

- Analyzing the relationship between the sum of squares of errors (SSE) and different K values, the elbow point on the SSE curve is the optimal K value.

Workflow of HydraLoRA

Finetuning: Define a set of experts, denoted as $(E_1, ..., E_N)$, consisted by multiple B matrices. A is the shared matrix. The forward process is expressed as :

$$y = W_0 x + \sum_{i=1}^{N} \omega_i E_i A x \quad (MoE)$$
$$\omega_i = softmax(W_g^T x) \quad (Router)$$

Inference: HydraLoRA merges adapters by enabling routing computation based on the input.

Outline

Dataset and Benchmarks

Single domain:

- **General:** Fine-tune with the general instruction tuning **databricks-dolly-15k** for generic language capability and evaluate with **MMLU**.
- **Medical:** Fine-tune with **GenMedGPT** and **clinic-10k** from ChatDoctor and evaluate medical tasks in **MMLU**.
- **Law:** Fine-tune with two legal instruction tuning datasets **Lawyer-Instruct** and **US-Terms** then evaluate with law tasks in **MMLU**.

Dataset and Benchmarks

Single domain:

- **Math:** Fine-tune with the training split of **GSM8K** for mathematical reasoning and evaluate with test set of **GSM8K**.
- **Code:** Fine-tune with **CodeAlpaca** for code generation and evaluate with **HumanEval**.

Multi-task domain:

- Fine-tune with a portion of **Flanv2** covering NLU and NLG, which can be grouped into 10 distinct tasks. Evaluate on **Big-Bench Hard** benchmark.

Baselines

Full Fine-tuning

PEFT methods

- **Prompt Tuning:** adds task-specific prompts to the input, and these prompt parameters are updated independently.
- **P-Tuning:** adds trainable prompt embeddings to the input that is optimized by a prompt encoder to find a better prompt.
- **Prefix Tuning:** prefixes a series of task-specific vectors to the input sequence that can be learned.

Baselines

PEFT methods

- **IA3:** infusing learned vectors into transformer architectures.
- **AdaLoRA:** more parameters are budgeted for important weight matrices and layers, while the less important ones receive fewer parameters.

Multiple LoRA weighted average methods

- **LoRA MoE**: BA is defined as an expert, and uses MOE to combine them.
- **LoraHub**: aggregates 20 LoRAs at random for new downstream tasks.

Overall Performance

Table 2: Comparative performance of different tuning schemes across multiple benchmarks on a single domain. 8-shot for GSM8K, zero-shot for others. $\#\bar{B}$ refers to the average B matrix number.

Schemes	MMLU	Medical	Law	HumanEval P@1	HumanEval P@10	GSM8K	%Param	#A	$\#\bar{B}$
LLaMA2-7B [46]	38.88	35.98	33.51	13.10	20.34	10.38	-	-	-
Full Fine-Tuning	49.91	46.78	46.08	20.24	32.93	25.70	100	-	-
Prompt Tuning [23]	39.91	37.59	35.02	13.66	21.55	13.18	0.001	-	-
P-Tuning$_{(256)}$ [28]	41.11	39.81	36.72	13.60	21.13	15.56	0.193	-	-
Prefix Tuning [24]	41.78	40.28	36.54	13.23	22.56	16.89	0.077	-	-
$(IA)^3$ [26]	40.45	37.12	35.25	13.54	23.17	13.98	0.009	-	-
AdaLoRA$_{(r=8)}$ [52]	44.32	42.83	39.36	14.81	23.78	19.51	0.093	1	1
LoRA$_{(r=8)}$ [17]	43.22	41.59	37.85	15.67	22.95	18.24	0.062	1	1
LoRA$_{(r=16)}$	45.45	43.10	39.64	16.71	25.60	20.32	0.124	1	1
LoRA$_{(r=32)}$	46.59	44.32	40.81	17.12	25.89	20.67	0.248	1	1
LoRA-Split$_{(4\times 8)}$	46.94	45.28	41.35	18.20	26.85	21.92	0.248	4	4
HydraLoRA$_{(r=8)}$	**47.22**	**45.71**	**42.18**	**18.31**	**27.43**	**22.27**	**0.124**	1	3

- It is more effective to use multiple smaller LoRA heads for specific tasks rather than one single LoRA for the entire domain dataset, given the same parameter count.
- Multiple LoRA heads, individually trained on different data, will improve efficiency by distinguishing matrix B parameters.

Overall Performance

Table 4: Comparative performance of different tuning schemes, including base model (Base), LoRA tuning (LoRA), LoraHub learning, multi-LoRA tuning with MoE inference (LoRA MoE) and our proposed *HydraLoRA* learning across mix-task domain on the BBH benchmark with LLaMA2-7B as the base LLM (3-shot).

Task	Base	LoRA	LoraHub	LoRA MoE	*HydraLoRA*
Boolean Expressions	61.9	67.1	72.9	68.0	73.7
Causal Judgement	52.2	54.9	50.1	51.4	53.2
Date Understanding	30.4	35.2	36.0	33.9	36.0
Disambiguation	34.8	45.2	49.1	47.2	50.3
Dyck Languages	15.8	18.7	14.5	16.8	19.8
Formal Fallacies	49.0	62.2	64.5	67.6	65.3
Geometric Shapes	9.7	17.7	18.7	17.7	19.7
Hyperbaton	51.8	74.3	74.3	68.9	77.2
Logical Deduction (five objects)	21.9	33.3	38.7	40.0	42.2
Logical Deduction (seven objects)	15.0	36.4	37.3	40.7	40.7
Logical Deduction (three objects)	32.8	41.4	38.5	43.7	42.9
Movie Recommendation	34.4	53.5	56.0	56.8	58.3
Multistep Arithmetic	1.2	1.2	1.9	1.9	1.8
Navigate	53.8	52.7	56.2	58.0	57.1
Object Counting	40.1	40.5	42.3	44.7	42.3
Penguins in a Table	21.7	23.2	25.0	23.2	25.9
Reasoning about Colored Objects	19.4	28.0	32.7	38.3	38.3
Ruin Names	24.3	28.7	34.3	34.3	36.7
Salient Translation Error Detection	11.3	11.1	17.1	16.2	20.1
Snarks	44.0	47.9	54.9	53.6	56.9
Sports Understanding	57.5	59.0	61.2	59.0	60.2
Temporal Sequences	21.1	32.6	28.9	34.1	30.4
Tracking Shuffled Objects (five objects)	21.9	23.7	23.7	28.0	29.3
Tracking Shuffled Objects (seven objects)	14.6	15.3	16.6	15.3	15.3
Tracking Shuffled Objects (three objects)	32.4	38.4	39.0	38.4	40.7
Web of Lies	51.4	52.8	53.2	50.1	52.0
Word Sorting	29.6	33.6	33.6	31.2	34.0
Avg Performance	31.6	36.8	39.7	40.3	41.5
# of A/B for training	0/0	1/1	48/48	48/48	1/10
# of A/B for inference	0/0	1/1	20/20	48/48	1/10
% Params	-	0.062	1.240	2.976	0.341

- HydraLoRA outperforms other merge methods in complex, multi-task domains, demonstrating superior scalability and robustness.

Overall Performance

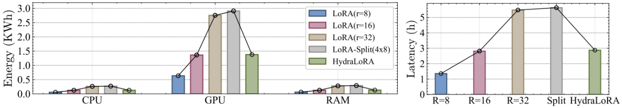


Figure 5: Energy consumption and latency during fine-tuning with different LoRA approaches (fine-tuning LLaMA2-7B with GSM-8K).

- HydraLoRA enhances the efficiency of the system, particularly in reducing training energy consumption and latency.

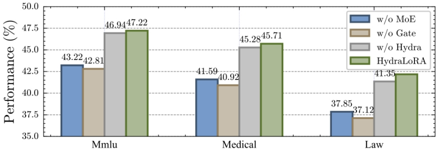


Figure 6: Comparative performance of ablation study for *HydraLoRA* across multiple benchmarks.

- The MoE architecture and the gate function are essential during the fine-tuning process.

Overall Performance

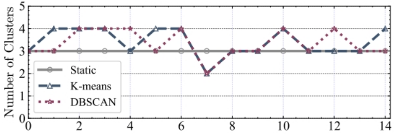


Figure 7: Number of clusters generated by different approaches including developer-specific (static), k-means, and DBSCAN.

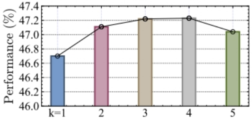


Figure 8: The results of experiments for hyper-parameters number of clusters.

- The K-means method has a similar performance to DBSCAN but with less complexity.
- The number k of clusters is NOT a sensitive parameter for HydraLora.

Outline

Conclusion

- This paper introduces a novel architecture HydraLoRA that features an asymmetric structure with a shared matrix for all samples and distinct matrices for each intrinsic component, aiming to adapt it to a new domain across various tasks.
- **Advantage:**
  1. The shared A and multiple B matrix structure enhances model performance across multi-domain, multi-task settings.
  2. It efficiently utilizes parameter distribution, resulting in much smaller parameters.
  3. The method for initializing the number of experts is insightful.
- **Disadvantage:** The addition of a router means the adapter weights don't fully integrate into the model, thus it has a higher inference latency than conventional methods.
- Conducting insightful exploratory experiments to observe and summarize patterns is essential in research.

Thank you!