UORA: Uniform Orthogonal Reinitialization Adaptation in Parameter Efficient Fine-Tuning of Large Models

Xueyan Zhang[1,*], **Jinman Zhao**[2,†,*], **Zhifei Yang**[3], **Yibo Zhong**[4], **Shuhao Guan**[5], **Linbo Cao**[1], **Yining Wang**[2]
[1]University of Waterloo, [2]University of Toronto,
[3]Peking University, [4]Independent Researcher, [5]University College Dublin

Abstract

This paper introduces Uniform Orthogonal Reinitialization Adaptation (UORA), a novel parameter-efficient fine-tuning (PEFT) approach for Large Language Models (LLMs). UORA achieves state-of-the-art performance and parameter efficiency by leveraging a low-rank approximation method to reduce the number of trainable parameters. Unlike existing methods such as LoRA and VeRA, UORA employs an interpolation-based reparametrization mechanism that selectively reinitializes rows and columns in frozen projection matrices, guided by the vector magnitude heuristic. This results in substantially fewer trainable parameters compared to LoRA and outperforms VeRA in computation and storage efficiency. Comprehensive experiments across various benchmarks demonstrate UORA's superiority in achieving competitive fine-tuning performance with negligible computational overhead. We demonstrate its performance on GLUE and E2E benchmarks and its effectiveness in instruction-tuning large language models and image classification models. Our contributions establish a new paradigm for scalable and resource-efficient fine-tuning of LLMs.

1 Introduction

Large models have demonstrated unparalleled performance in many tasks (Li, 2024; Zhang et al., 2025; Wu et al., 2025a; Ma et al., 2025). Popular models like GPT-3 (Brown et al., 2020), LLaMA3 (Dubey et al., 2024), and Qwen (Yang et al., 2024a) have shown promising capability in various practical domains such as Reasoning (Cai et al., 2025; Yang et al., 2024b; Xu et al., 2025; Li et al., 2025b; Bi et al., 2025b), education (Chen et al., 2025) and RAG (Peng et al., 2024a; Liu et al., 2024a; Peng et al., 2024b). However, many downstream applications still require fine-tuning to improve the zero-shot experience, incorporate domain-specific content (Wu et al., 2025b; Zheng et al., 2025; Luo et al., 2025; Wang et al., 2025; Dong et al., 2024a), and optimize performance in diverse scenarios (Chen et al., 2024; Yu et al., 2025). The full fine-tuning (FFT) process adapts the model by updating all pre-trained weights within the model. Depending on the goal of downstream tasks, FFT may require a significant amount of time and computing resources; meanwhile, as the model size increases exponentially, the demand for multiple checkpoints for iterative pre-training and downstream adaption cycles poses significant challenges to storage efficiency.

Parameter-efficient fine-tuning (PEFT) methods aim to address this challenge by injecting adapters for the fine-tuning process while keeping the pre-trained weights untouched. Compared to the existing approaches at the time (Houlsby et al., 2019; Rebuffi et al., 2017; Li and Liang, 2021; Lester et al., 2021; Hambardzumyan et al., 2021; Liu et al., 2024b), Low-Rank Adaptation (LoRA) exploits the low intrinsic dimensionality in weight matrices and performs low-rank approximation with a pair of projection-down and projection-up matrices; LoRA uses significantly less trainable parameter and thus largely improves the computing and storage efficiency (Hu et al., 2021). As a variant method, Vector-based Random Matrix Adaptation (VeRA) freezes the two low-rank matrices via randomization and introduces two trainable scaling vectors, which again reduces the number of trainable parameters (Kopiczko et al., 2023). As a result of randomized frozen matrix initialization, VeRA requires a higher rank for comparable performance, which increases the computation complexity. AFLoRA points out that any reduction in VeRA's rank causes a drop in fine-tuning performance (Liu et al., 2024c).

Extending the state-of-the-art, we propose Uniform Orthogonal Reinitialization Adaptation

*Equal contribution
†Corresponding author: jzhao@cs.toronto.edu

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 11709–11728
July 27 - August 1, 2025

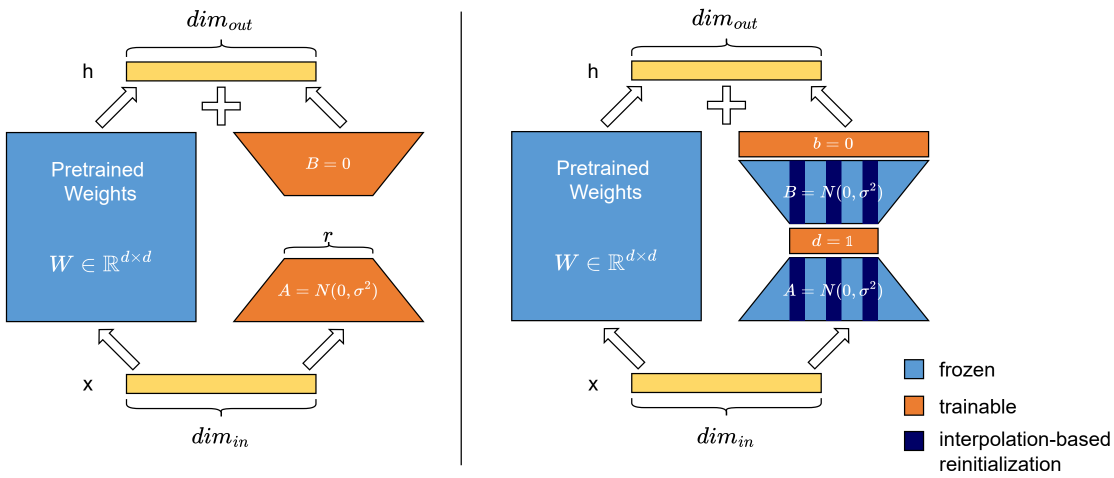

Figure 1: **Overview of LoRA (left) and UORA (right).** LoRA trains a pair of projection matrices, namely A and B, with low rank r. The update to the pretrained weights is thus represented as $\Delta W = A \times B$. UORA adopts the similar strategy as VeRA; both projection matrices are frozen and randomized. A pair of scaling vectors, $\vec{d}$ and $\vec{b}$, is trained to adapt the frozen matrices. The key difference is that UORA applies interpolation reinitialization mechanism to selectively and partially update A and B. Similar to all LoRA-based PEFT methods, the learned weight update ΔW could be merged into W for zero inference latency.

(UORA), a PEFT method that uses substantially few parameters than LoRA while showing close performance on various fine-tuning tasks. Building upon VeRA, UORA also adopts a pair of randomized frozen projection matrices and a pair of scaling vectors $\vec{d}$ and $\vec{b}$ for adaptation. We introduce an interpolation-based reparametrization mechanism: during the training process, UORA selectively reinitializes a part of the frozen matrices. UORA achieves comparable or better performance than VeRA with a much lower rank. With this approach, there is no need to perform gradient update on the projection matrices and the only trainable parameters are the pair of adaptation vectors. UORA achieves state-of-the-art in terms of the number of trainable parameters and demonstrates efficient fine-tuning performance across various benchmarks. Compared with LoRA, UORA achieves competitive fine-tuning performance while demanding 15x and 8x fewer parameters in GLUE and E2E benchmarks respectively.

Our main contributions are as follows:

- We propose a novel parameter-efficient fine-tuning[1] method with no inference latency, Uniform Orthogonal Reinitialization Adaptation. Extending the state-of-the-art PEFT methods, UORA achieves performant results with much less trainable parameters.
- We compare our approach with LoRA, VeRA and other PEFT methods on natural language understanding (GLUE), and natural language generation (E2E) benchmarks, and compare instruction-tuning and image classification tasks.
- We conduct ablation studies to gain insights on various components of our approach and their impact on performance and efficiency.

2 Related Work

2.1 Low-Rank Adaptation

The state-of-the-art LoRA explores the intrinsic low dimension in language models, which involves finding a pair of low-rank decomposition matrices that capture the essential information of the weight updates during the training process (Hu et al., 2021). The decomposition contains a pair of projection-down and projection-down matrices, which significantly reduced the number of trainable parameters, and thus it largely increases the training and storage efficiency. Additionally, these delta weights could be merged into the original model weights leading to zero inference latency.

A collection of methods extends the LoRA concept. Dynamic LoRA (DyLoRA) allows for a range

[1]Code available at: https://github.com/zhaojinm/UORA

of training ranks, rather than a fixed one, to avoid the exhaustive search for the optimal rank (Valipour et al., 2022). Adaptive LoRA (AdaLoRA) adjusts the rank adaptively across different layers within a given budget (Zhang et al., 2023a).

Vector-Based Random Matrix Adaptation (VeRA) freezes the pair of projection matrices and introduces trainable scaling vectors for adaptation (Kopiczko et al., 2023). AFLoRA reduces the decomposition rank by adaptively freezing the projection matrices at the beginning and eventually uses scaling vectors as VeRA (Liu et al., 2024c). Our approach falls under the LoRA category and improves upon VeRA.

Some research streams explore matrices decomposition into smaller chunks (Shen et al., 2024; Ren et al., 2024), while others (Shi et al., 2024) add residue modules. Dou et al. (2024); Zhang et al. (2024) embed Mixture of Experts(MoE) architecture into LoRA. There are also various LoRA-based works (Tian et al., 2024; Ding et al., 2023; Kim et al., 2024). Quantization techniques have also been added to further decrease memory usage (Dettmers et al., 2023; Xu et al., 2024; Guo et al., 2024).

2.2 Parameter-Efficient Fine-Tuning

PEFT refers to a series of methodologies that aim to achieve comparable fine-tuning performance at reduced costs in terms of time and storage requirements. Compared to full fine-tuning, PEFT demonstrates advantages in low to medium resource setups and could be outperformed by full fine-tuning in high-resource setup (Naveed et al., 2023; Chen et al., 2022). In other words, depending on the location of the adaptation layers, PEFT methods are roughly divided into two categories, non-weight-based and weight-based.

Adapter tuning injects one or more trainable layers into the transformer blocks sequentially or in parallel (He et al., 2021; Houlsby et al., 2019; Wang et al., 2022; Fang et al., 2023); an adapter layer usually consists of downscaling, non-linearity and upscaling. The bottleneck with adapter tuning is that it introduces latency during inference and may reduce the parallelism in GPU computing operations.

Prompt tuning concatenates trainable prompt parameters withel embeddings for downstream tasks, and its derivatives address limitations in training instability and forgetting (Liu et al., 2024b; Lester et al., 2021; Liu et al., 2021; Jin et al., 2024).

Prefix tuning places a set of trainable vectors in the frozen transformer layers (Li and Liang, 2021; Zhang et al., 2023b). Prefix tuning shows non-linearity in performance against the number of trainable parameters.

Bias tuning, specifically BitFit, focuses on training the bias, a subset of the total trainable parameters in the model; however, it may appear limited when the training data are large (Zaken et al., 2021).

There are some other efficient techniques such as data/token selection (Dong et al., 2024b; Hu et al., 2024; Sun et al., 2025), knowledge editing (Deng et al., 2025; Feng et al., 2025), model pruning/compression (Yu et al., 2024; Zhou et al., 2023), distillation (Feng et al., 2024; Jia, 2024; Wang et al., 2023), Linear Representation-Steering (Bi et al., 2025a),chunk-wise gradient computation (Li et al., 2025a) and fine-tune partial layers of the model (Fan et al., 2025).

2.3 Parameter Modeling

It has been widely studied to model the parameter distribution using randomized matrices for efficiency. Given the model sparsity, randomly initialized weight matrices in neural networks appears to contain high performance sub-networks that requires less or even no training (Frankle and Carbin, 2018; Ramanujan et al., 2020). It is empirically demonstrated that language models indeed possess low intrinsic dimension and it is efficient to employ randomized projection matrices (Aghajanyan et al., 2020). Meanwhile, there is a thread of research studying how information is distributed within LMs, and some works suggest that different neurons store different types of information (Niu et al., 2022, 2024).

3 Methodology

In this section, we present Uniform Orthogonal Reinitialization Adaptation (UORA), a novel parameter-efficient fine-tuning approach inspired by state-of-the-art PEFT methods.

3.1 Method Formulation

LoRA exploits the low-intrinsic dimensionality in weight matrices and reparameterizes the weight update in fine-tuning with a pair of low-rank projection matrices. Formally, for a pretrained weight matrix $W_0 \in \mathbb{R}^{d\times k}$, the weight updates after fine-tuning are decomposed by $\Delta W = BA$, where

Models	Rank	LoRA # Trainable Parameters	LoRA Required Bytes	VeRA / UORA # Trainable Parameters	VeRA / UORA Required Bytes
RoB_{base}	1	36.8K	144KB	18.4K	72KB
	16	589.8K	2MB	18.8K	74KB
	256	9437.1K	36MB	24.5K	96KB
$\text{RoB}_{\text{large}}$	1	98.3K	384KB	49.2K	192KB
	16	1572.8K	6MB	49.5K	195KB
	256	25165.8K	96MB	61.4K	240KB
GPT-3	1	4.7M	18MB	2.4M	9.1MB
	16	75.5M	288MB	2.8M	10.5MB
	256	1207.9M	4.6GB	7M	33MB

Table 1: Theoretical memory required to store trained LoRA, VeRA, and UORA weights for RoBERTa Base (RoB_{base}), and Large ($\text{RoB}_{\text{large}}$), and GPT-3 models. We assume that all methods are applied on query and key layers of individual attention modules. Although UORA shares the same number of parameter for a given rank, UORA requires a lower rank by orders of magnitudes than VeRA in almost all scenarios.

$B \in \mathbb{R}^{d\times r}, A \in \mathbb{R}^{r\times k}, r \ll min(d,k)$. The forward pass is then modeled as:

$$h = W_0x + \Delta Wx = W_0x + BAx, \quad (1)$$

To improve parameter efficiency, VeRA freezes the projection matrices A and B using randomized initialization, and introduces trainable, scaling vectors $\vec{d}$ and $\vec{b}$:

$$h = W_0x + \Delta Wx = W_0x + \Lambda_b B \Lambda_d Ax, \quad (2)$$

Λ_d and Λ_b are the scaling vectors; they effectively scale or disable rows and columns in frozen matrices A and B. Although the number of trainable parameters decreases, a higher rank is usually required for on-par fine-tuning performance.

UORA adopts the weight update formula from Equation 2 in VeRA. However, it introduces a weight reinitialization mechanism, which has been shown to be highly efficient in neural network training (Zaidi et al., 2023). After each iteration, we examine each entry in the scaling vector. If its magnitude falls below a threshold τ for a consecutive count of k times, the corresponding column in matrix A and the corresponding row in matrix B are updated. To smooth the training process and avoid inconsistent performance drops, linear interpolation (LERP) with a factor α is applied between the old and new values. The interpolation-based reinitialization is achieved through the following formula. :

$$v_{rand} = Random(v_{old}) \quad (3)$$

$$v_{new} = \alpha v_{old} + (1-\alpha)v_{rand} \quad (4)$$

3.2 Details

Figure 1 right panel illustrates the pipeline of UoRA.

Orthogonal Uniform Initialization. The impact of initialization methods is studied in VeRA. The initialization method preserves matrix expressivity while maintaining a well-conditioned weight space for parameter modeling. However, in our work, we choose to use orthogonal initialization, based on its ability to improve gradient flow, ensure stability in deep networks, and accelerate convergence by preserving variance across layers (Hu et al., 2020; Huang et al., 2021).

Dimension Pruning. Although the number of trainable parameters drastically reduces, VeRA heavily replies on the randomly initialized frozen matrices for reparameterization; thus it must adopt higher ranks for competitive performance in most cases. Rank 1024 in VeRA is commonly seen as oppose to rank 32 in LoRA. Higher ranks punish not only the computation efficiency due to larger matrix multiplications but also storage efficiency.

The magnitude of scaling vectors indicates how important of the corresponding column or row in the frozen matrix A and B (Sun et al., 2024). Given a dimension in $\vec{d}$ whose magnitude is insignificant

($|\vec{d}|$ below threshold τ towards 0), then this dimension contributes less or even negligible to the weight update ΔW. Thus, a lower rank could be used for efficient reparameterization.

Interpolation-based Reinitialization. To enhance stability and prevent newly generated weights from becoming excessively large, we applied linear interpolation. This interpolation smooths the frozen weight updates, ensuring that the new weights do not deviate too far from the old ones; thus avoiding extreme values that could negatively impact training progress and consequently model performance. Linear interpolation is also widely used in machine learning (Eysenbach et al., 2024) and can be used to control the magnitude of parameter updates and prevent instability during training (Iyer et al., 2024; Berrada et al., 2020).

3.3 Parameters

Following the naming convention in the prior work, the number of trainable parameters in UORA is then

$$L_{tuned} \times (d_{model} + r). \tag{5}$$

where r denotes the rank, d_{model} denotes the dimension of a given layer and L_{tuned} denotes the number of fine-tuned layers. VeRA share the same equation despite that its rank r is typically larger the UORA's in most cases. The number of trainable parameters in LoRA is computed as

$$2 \times L_{tuned} \times d_{model} \times r \tag{6}$$

Benefied from the interpolation-based reinitialization mechanism, UORA can adopt a rank r close to LoRA and much lower than VeRA; Table 1 compares the parameter efficiency in RoBERTa and GPT-3 models.

4 Experiments

In this section, we perform evaluations of UORA in the diverse domains, covering natural language understanding (NLU), natural language generation (NLG), instruction-tuning, and computer vision (CV). At last, we present an ablation study that sheds light on the effect of each component in our method.

Baselines. We compare UORA method with the state-of-the-art parameter-efficient fine-tuning methods. For fair, comprehensive comparison, we extend the experiment settings from prior works as much as we could. Baselines include:

- Full fine-tuning (FFT) - All pretrained parameters in the model subjects to gradient updates for a downstream task; it offers the maximum flexibility but requires considerable resources.
- Adapter tuning - Following the naming convintion, $\text{Adapter}^{\text{H}}$ injects adapters, with two fully connected layers and activation functions, between the attention module, multi-layer perceptron (MLP) module and the following residual connection (Houlsby et al., 2019). $\text{Adapter}^{\text{L}}$ reduces the number of parameters by injecting only after MLP module and layer normalization operation (Lin et al., 2020). Similarly, $\text{Adapter}^{\text{P}}$ (Adapterfusion) injects the adapter layer after the feed-forward network (FFN) module (Pfeiffer et al., 2020). $\text{Adapter}^{\text{D}}$, also known as AdapterDrop, selectively drops adapter layers for parameter efficiency (Rücklé et al., 2020).
- Bitfit - Bias tuning focuses on training bias vectors while freezing the rest of pretrained parameters in the model. It may appear less efficient when training data is large (Zaken et al., 2021).
- LoRA - The state-of-the-art PEFT method that reparameterize the weight updates in form of $\Delta W = BA$, where $B \in \mathbb{R}^{d \times r}, A \in \mathbb{R}^{r \times k}$ and rank r is much smaller than model dimension for better paramter efficiency (Hu et al., 2021).
- AdaLoRA - Adaptive Low-Rank Adaptation lift the need to search for the rank r via budget allocation strategy by employing singular value decomposition (SVD) adaptation (Zhang et al., 2023a).
- VeRA - Vector-based Random Matrix Adaptation further reduces the number of trainable parameters by freezing the projection matrices. It introduces a pair of trainable vectors, b and d for adaptation (Kopiczko et al., 2023).
- Red - Representation EDiting directly operates on representations at some layers via scaling and biasing operations (Wu et al., 2024).

4.1 Natural Language Understanding

We evaluate UORA method on the General Language Understanding Evaluation (GLUE) (Liu, 2019) benchmark, a collection of natural language understanding (NLU) tasks. Typical tasks covers grammar, semantics, inference, and paraphrasing (Wang, 2018).

Experiment Details. We employ RoBERTa Base (RoB_{base}) and RoBERTa Large ($\text{RoB}_{\text{large}}$) for the

Model & Method	# Trainable Parameters	SST-2 (Acc.)	MRPC (Acc.)	CoLA (MCC)	QNLI (Acc.)	RTE (Acc.)	STS-B (PCC)	Avg.
RoB_{base}(FF)	125M	94.8	90.2	63.6	92.8	78.7	91.2	85.2
RoB_{base}(BitFit)	0.1M	93.7	92.7	62.0	91.8	81.5	90.8	85.4
RoB_{base}(Adpt^{D})	0.3M	$94.2_{\pm0.1}$	$88.5_{\pm1.1}$	$60.8_{\pm0.4}$	$93.1_{\pm0.1}$	$71.5_{\pm2.7}$	$89.7_{\pm0.3}$	83.0
RoB_{base}(Adpt^{D})	0.9M	$94.7_{\pm0.3}$	$88.4_{\pm0.7}$	$62.6_{\pm0.6}$	$93.0_{\pm0.2}$	$75.9_{\pm2.2}$	$90.3_{\pm0.4}$	84.2
RoB_{base}(LoRA)	0.3M	$95.1_{\pm0.2}$	$89.7_{\pm0.7}$	$63.4_{\pm1.2}$	$93.3_{\pm0.3}$	$86.6_{\pm0.7}$	$91.5_{\pm0.2}$	85.2
RoB_{base}(AdaLoRA)	0.3M	$94.5_{\pm0.2}$	$88.7_{\pm0.6}$	$62.0_{\pm0.4}$	$93.1_{\pm0.2}$	$81.0_{\pm0.6}$	$90.5_{\pm0.2}$	85.0
RoB_{base}(VeRA)	0.043M	$94.6_{\pm0.1}$	$89.5_{\pm0.5}$	$65.6_{\pm0.8}$	$91.8_{\pm0.2}$	$78.7_{\pm0.7}$	$90.7_{\pm0.2}$	85.2
RoB_{base}(RED)	0.02M	$93.9_{\pm0.3}$	$89.2_{\pm1.0}$	$61.0_{\pm3.0}$	$90.7_{\pm0.4}$	$78.0_{\pm2.1}$	$90.4_{\pm0.3}$	83.9
RoB_{base}(UORA)	0.019M	$94.2_{\pm0.5}$	$90.4_{\pm0.5}$	$65.4_{\pm0.5}$	$91.2_{\pm0.2}$	$87.1_{\pm0.8}$	$90.6_{\pm0.2}$	**86.5**
$\text{RoB}_{\text{large}}$(FF)	356M	96.4	90.9	68.0	94.7	86.6	92.4	88.2
$\text{RoB}_{\text{large}}$($\text{Adpt}^{\text{P}}$)	3M	$96.1_{\pm0.3}$	$90.2_{\pm0.7}$	$68.3_{\pm1.0}$	$94.8_{\pm0.2}$	$83.8_{\pm2.9}$	$91.9_{\pm0.4}$	87.6
$\text{RoB}_{\text{large}}$($\text{Adpt}^{\text{P}}$)	0.8M	$96.6_{\pm0.2}$	$89.7_{\pm1.2}$	$67.8_{\pm2.5}$	$94.8_{\pm0.3}$	$80.1_{\pm2.9}$	$91.9_{\pm0.4}$	86.8
$\text{RoB}_{\text{large}}$($\text{Adpt}^{\text{H}}$)	6M	$96.2_{\pm0.3}$	$88.7_{\pm2.9}$	$66.5_{\pm4.4}$	$94.7_{\pm0.2}$	$83.4_{\pm1.1}$	$91.9_{\pm0.7}$	86.9
$\text{RoB}_{\text{large}}$($\text{Adpt}^{\text{H}}$)	0.8M	$96.3_{\pm0.2}$	$87.7_{\pm1.7}$	$66.3_{\pm2.7}$	$94.8_{\pm0.3}$	$72.9_{\pm2.9}$	$91.7_{\pm0.4}$	84.9
$\text{RoB}_{\text{large}}$(LoRA)	0.8M	$96.2_{\pm0.5}$	$90.0_{\pm1.0}$	$68.2_{\pm1.9}$	$94.8_{\pm0.3}$	$85.2_{\pm1.2}$	$92.3_{\pm0.5}$	87.8
$\text{RoB}_{\text{large}}$(VeRA)	0.061M	$96.1_{\pm0.1}$	$90.9_{\pm0.7}$	$68.0_{\pm0.8}$	$94.4_{\pm0.2}$	$85.9_{\pm0.7}$	$91.7_{\pm0.8}$	87.8
$\text{RoB}_{\text{large}}$(UORA)	0.049M	$96.1_{\pm0.2}$	$92.2_{\pm0.5}$	$69.3_{\pm0.5}$	$94.5_{\pm0.5}$	$87.0_{\pm1.5}$	$91.7_{\pm0.5}$	**88.5**

Table 2: GLUE benchmark performance on RoBERTa Base (RoB_{base}) and RoBERTa Large ($\text{RoB}_{\text{large}}$). We report Matthew's correction coefficient (MCC) for CoLA, Pearson correlation coefficient (PCC) for STS-B, and accuracy (Acc.) for the rest tasks. Results for prior works are taken from Gao et al. (2024), Kopiczko et al. (2023) and Wu et al. (2024).

GLUE benchmark. With the aforementioned interpolation-based reinitialization methodology, UORA now can employ a much smaller rank r than VeRA. For instance, UORA adopts a rank of 16 for MRPC task while VeRA requires a rank of 1024. Detailed rank for individual tasks can be found in Appendix A. We use orthogonal uniform initialization for both projection matrices A and B and we initialize the vector $\vec{d}$ and $\vec{b}$ with 0.1 and 0 respectively.

Following prior work, UORA is applied to query and value projection matrices in the transformer blocks. Similar to VeRA, we use separate learning rate for the head and UORA layers to mitigate the additional scaling hyperparameter introduced in LoRA. Refer to Table 6 in Appendix A for detailed hyperparameter settings.

We omit MNLI and QQP tasks in GLUE benchmark due to budget limitations. We report the average of 5 runs with the best epoch evaluation result using randoms seeds in Appendix A.

Results. Table 2 shows the results of UORA and baselines on GLUE benchmark. UORA demonstrates competitive performance compared to LoRA while using approximately 15x less parameters. With an interpolation-based reinitialization mechanism, UORA is able to improve the performance on most of tasks compared to VeRA.

4.2 E2E Benchmark

The E2E NLG benchmark, released by Novikova et al. (2017), is an English dataset verbalized a set of 2-9 key-value attribute pairs in the restaurant domain, with more than 51K combinations of dialogues.

Experiment Details. For natural language generation task, we evaluate UORA and other PEFT methods following the setup described in LoRA (Hu et al., 2021). We fine-tune GPT-2 Medium and Large models (Radford et al., 2019) on the E2E benchmark dataset. We adopt a rank of 32 for LoRA and UORA, 1024 for VeRA; the rest baselines are extended from prior work. We apply hyparameter tuning and details are depicted in the Appendix A.

Results. As we can see from Table 3, UORA shows competitive performance compared to LoRA and VeRA, whose number of trainable parameters are 8x and 2x more. Using similar number of parameters, UORA shows dominating performance in all tasks compared to RED.

Model	Method	# Trainable Parameters	BLEU	NIST	METEOR	ROUGE-L	CIDEr
GPT2 Medium	FF*	354.92M	65.95	8.52	45.95	69.13	2.35
	Adpt^H*	0.9M	64.31	8.29	44.91	67.72	2.28
	Adpt^P*	0.8M	64.41	8.30	44.74	67.53	2.29
	LoRA	0.4M	67.14	8.65	46.05	69.50	2.41
	VeRA	0.098M	66.34	8.52	45.84	69.39	2.39
	RED	0.050M	64.62	8.33	45.14	67.46	2.25
	UORA	0.051M	66.67	8.62	45.37	68.82	2.35
GPT2 Large	FT*	774.03M	65.56	8.50	45.40	68.38	2.27
	Adpt^H*	1.8M	65.94	8.46	45.78	68.65	2.23
	Adpt^P*	1.5M	65.53	8.41	45.65	68.46	2.33
	LoRA	0.77M	68.07	8.74	46.28	69.92	2.43
	VeRA	0.18M	66.97	8.59	46.07	69.37	2.41
	RED	0.09M	65.22	8.40	45.59	68.14	2.34
	UORA	0.14M	67.77	8.60	46.17	69.05	2.40

Table 3: Performance of our methods on E2E test set via GPT2-medium and GPT2-large. Results with * are taken from Wu et al. (2024), rest are replicated by ourselves.

4.3 Instruction Tuning

Pretrained models possess latent capabilities, which fine-tuning can unlock for specific downstream tasks. Building upon fine-tuning, instruction tuning introduces an additional instruction component into the training data, enabling models to better follow task-specific directives. As a result, the instruction-tuned models become more robust and versatile in addressing prompt questions even with better reasoning capability. To demonstrate UORA's generalization across various models and its effectiveness on more complex tasks, we conduct evaluation on **Arithmetic Reasoning** benchmarks.

The evaluation dataset covers a wide range of arithmetic reasoning problems in various formats: 1) the **AddSub** dataset contains simple math word problems on addition and subtraction, typically involving one-step arithmetic reasoning (Hosseini et al., 2014); 2) the **MultiArith** dataset is a collection arithmetic word problems that require multiple operation steps like addition, subtraction, multiplication, and division to reach the solution (Roy and Roth, 2015); 3) the **SVAMP** dataset consists of arithmetic word problems up to grade 4 level, which is obtained by modifying existing datasets to reduce annotation artifacts (Patel et al., 2021); 4) the **SingleEq** dataset covers problems that are solvable using a single linear equation, emphasizing the translation from natural language to algebraic expressions (Koncel-Kedziorski et al., 2015).

Experiment Details. We select LLaMA 7B and LLaMA 3-8B as the representative pre-trained models for instruction tuning. For reproducibility purposes, we extend the experiment setting from previous work (Hu et al., 2023); models are fine-tuned on the high-quality Math10K dataset, obtained by selecting examples from arithmetic reasoning datasets and adding step-by-step rationales. The fine-tuned models are then evaluated on AddSub, MultiArith, SVAMP, and SingleEq dataset.

All methods, LoRA, VeRA and UORA, are applied to all attention layers (namely W_Q, W_K, W_V, W_o). A rank of 8 for LoRA is used, a rank 1024 for VeRA and rank 32 for UORA. Detailed hyperparameter settings are disclosed in Appendix A.

Results. Table 4 shows similar pattern observed in previous experiments. We find that UORA maintains competitive performance in arithmetic reasoning in both LLaMA 7B and LLaMA 3-8B, despite using a minimal number of parameters shown in the column # Trainable Parameters. The UORA's extreme parameter efficiency could facilitate resource-constrained scenarios.

LoRA, with the most number of parameters and maximum flexibility, shows strong performance and sets the baseline for the rest methods. Despite using orders of magnitude fewer parame-

Model	Method	# Trainable Parameters	AddSub	MultiArith	SVAMP	SingleEq
LLaMA 7B	LoRA	8.4M	0.8051	0.9383	0.4286	0.7715
	VeRA	0.59M	0.8069	0.9300	0.3944	0.7303
	UORA	0.50M	0.8095	0.9133	0.4210	0.7047
LLaMA 3-8B	LoRA	8.4M	0.8962	0.9633	0.7539	0.9665
	VeRA	0.59M	0.8886	0.9383	0.7027	0.9409
	UORA	0.50M	0.8911	0.9750	0.7100	0.9688

Table 4: Instruction tuning performance on arithmetic reasoning tasks using LLaMA 7B and LLaMA 3-8B models.

ters, UORA closely follows LoRA in performance (e.g., in AddSub, LoRA scores 89.62% and UORA closely follows with 89.11% on LLaMA3-8B).

We are inspired by VeRA, which serves as our most immediate baseline; VeRA's scaling vectors are solely used to enable entries in frozen matrices. UORA, in contrast, could potentially offer better performance, because the interpolation-based reinitialization enhances the expressiveness of the frozen matrices. This not only reduces the number of trainable parameters but also offers improved fine-tuning performance as shown in Table 4.

4.4 Image Classification

PEFT methods are as efficient in fine-tuning for image classification tasks. Vision Transformer (ViT) (Dosovitskiy et al., 2021) successfully extends the attention mechanism to address image classification tasks. In this section, we evaluate our method in ViT on common datasets.

Experiment Details. For evaluation, we adopt both ViT base and large models, pretrained on ImageNet-21K, on a wide range of datasets, including RESIS45 (Cheng et al., 2017), CIFAR100 (Krizhevsky et al., 2009), FOOD101 (Bossard et al., 2014), FLOWER102 (Nilsback and Zisserman, 2008).

We include full fine-tuning and training classification head only as baselines, as well as LoRA and VeRA. Specifically, we apply all methods on the query and value layers in ViT models. We followed the convention to use rank 8 for LoRA, 256 for VeRA and 32 for UORA. Each run has 10 epochs, and detailed hyperparameter settings are listed in Appendix A.

Results. As seen in Table 5, UORA is able to outperform VeRA and achieve comparable performance to LoRA with fewer parameters. For example, on CIFAR100, UORA reaches 96.08% accuracy with ViT-B and 95.72% with ViT-L, outperforming VeRA in both cases and approaching LoRA's performance, despite the number of trainable parameters is only 19.2K and 50.7K respectively. This demonstrates the efficiency and generalization of UORA as a PEFT method for vision tasks.

4.5 Ablation Study

4.5.1 Scalability

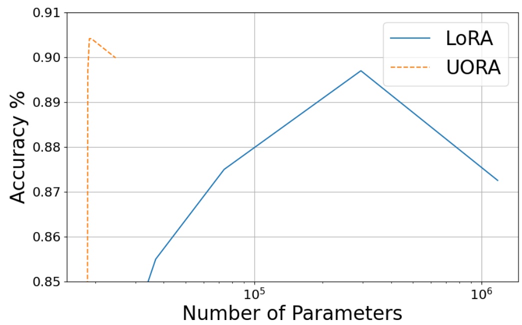

Figure 2: Performance vs. number parameters of LoRA and UORA on MPRC in GLUE benchmark.

We model the relation between fine-tune performance and the number of trainable parameters using RoBERTa Large model on MRPC in GLUE benchmark. We adopt rank $r = \{1, 2, 4, 16, 64\}$ for LoRA and $r = \{1, 4, 16, 32, 256\}$ for UORA. As shown in Figure 2, UORA demonstrates superior parameter efficiency, achieving comparable performance to LoRA while requiring fewer trainable parameters. In higher ranks, both UORA and LoRA show performance slight degradation, which could be a sign for excessive amount of parameters.

4.5.2 Impact of Initialization Method

As aforementioned, we employ orthogonal uniform initialization for frozen projection matrices A and B, as opposed to the Kaiming initialization in VeRA. The randomized projection matrices

Model	Method	# Trainable Parameters	CIFAR100	Food101	Flowers102	RESISC45
ViT-B	Head	–	79.02	70.85	56.86	40.84
	Full	85.8M	95.00	85.25	98.05	93.28
	LoRA	294.9K	96.78	86.49	99.09	94.29
	VeRA	24.6K	95.38	84.78	98.03	92.01
	UORA	19.2K	96.08	83.84	99.02	92.43
ViT-L	Head	–	82.22	72.63	63.26	50.47
	Full	303.3M	95.44	88.64	99.02	95.23
	LoRA	786.4K	96.56	85.38	99.00	94.60
	VeRA	61.4K	95.78	85.08	98.04	93.54
	UORA	50.7K	95.72	84.07	98.04	93.75

Table 5: Vision transformer results on image classification datasets with ViT Base (Vit-B) and Large (ViT-L). We report accuracy (%) after 10 epochs.

are shared across all target layers. The scaling vector $\vec{d}$ is initialized to 0.1 for efficient training as proved in VeRA, and and $\vec{b}$ is set to 0 such that the first forward pass is not affected, e.g., $\Delta W = \Lambda_b B \Lambda_d A x = 0$.

We conducted experiment to investigate the impact of initialization methods. We implement different matrix initialization methods on top of VeRA (equivalent to UORA with reinitialization disabled) and test the performance for MRPC task in GLUE bench mark. Table 10 shows that both xavier and orthogonal uniform initialization shows slightly better performance than Kaiming uniform; random uniform intialization appears to be less effective in parameter modeling given its inefficacy in performance, which aligns with the observation in the VeRA paper.

4.5.3 Impact of Threshold τ, Count k, and Interpolation Factor α

The frequency of interpolation-based reinitialization depends on the threshold τ and the count k. The extent of reinitialization depends on the interpolation factor α. After empirical study, we found that even though (τ, k, α) are correlated, the hyperparameter tuning can be simplified; we depict practical tuning strategy in A.1. We conduct detailed ablation study in Appendix B.

5 Conclusion

In this paper, we introduce UORA, a novel PEFT method achieving both fine-tuning performance and parameter efficiency. We extend VeRA and propose a simple yet effective interpolation-based reinitialization mechanism to improve the reparameterization efficiency. Our evaluation validates that UORA is applicable to various domains including NLP and CV tasks. Compared to the state-of-the-art LoRA, UORA demands 15x and 8x fewer parameters while demonstrating competitive performance in GLUE and E2E benchmarks respectively.

6 Limitations

We demonstrate the performance of our methods on various application domains. There are many directions for future explorations. While UORA updates the frozen matrices for improved performance, more research could be done for more effective parameter modeling (e.g., guided by fine-grained heuristics). Altough UORA demonstrates competitive performance, there are remaining potentials to fuse with other PEFT methods (e.g., alleviating the search of rank with singular value decomposition). Applying PEFT methods on extremely large language models, especially multimodal model, remains an open challenge.

References

Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. 2020. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13255*.

Leonard Berrada, Andrew Zisserman, and M Pawan Kumar. 2020. Training neural networks for and by interpolation. In *International conference on machine learning*, pages 799–809. PMLR.

Jinhe Bi, Yujun Wang, Haokun Chen, Xun Xiao, Artur Hecker, Volker Tresp, and Yunpu Ma. 2025a. Llava steering: Visual instruction tuning with 500x fewer parameters through modality linear representation-steering. *Preprint*, arXiv:2412.12359.

Jinhe Bi, Danqi Yan, Yifan Wang, Wenke Huang, Haokun Chen, Guancheng Wan, Mang Ye, Xun Xiao, Hinrich Schuetze, Volker Tresp, and Yunpu Ma. 2025b. Cot-kinetics: A theoretical modeling assessing lrm reasoning process. *Preprint*, arXiv:2505.13408.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. 2014. Food-101–mining discriminative components with random forests. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part VI 13*, pages 446–461. Springer.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Chengkun Cai, Xu Zhao, Yucheng Du, Haoliang Liu, and Lei Li. 2025. t^2 of thoughts: Temperature tree elicits reasoning in large language models. *Preprint*, arXiv:2405.14075.

Guanzheng Chen, Fangyu Liu, Zaiqiao Meng, and Shangsong Liang. 2022. Revisiting parameter-efficient tuning: Are we really there yet? *arXiv preprint arXiv:2202.07962*.

Jiali Chen, Xusen Hei, Yuqi Xue, Yuancheng Wei, Jiayuan Xie, Yi Cai, and Qing Li. 2024. Learning to correction: Explainable feedback generation for visual commonsense reasoning distractor. In *Proceedings of the 32nd ACM International Conference on Multimedia*, MM '24, page 8209–8218. ACM.

Jiali Chen, Xusen Hei, Yuqi Xue, Zihan Wu, Jiayuan Xie, and Yi Cai. 2025. Classic4Children: Adapting Chinese literary classics for children with large language model. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 2473–2488, Albuquerque, New Mexico. Association for Computational Linguistics.

Gong Cheng, Junwei Han, and Xiaoqiang Lu. 2017. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883.

Jingcheng Deng, Zihao Wei, Liang Pang, Hanxing Ding, Huawei Shen, and Xueqi Cheng. 2025. Everything is editable: Extend knowledge editing to unstructured data in large language models. In *The Thirteenth International Conference on Learning Representations*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient finetuning of quantized LLMs. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. 2023. Sparse low-rank adaptation of pre-trained language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4133–4145, Singapore. Association for Computational Linguistics.

Junhao Dong, Yuan Wang, Xiaohua Xie, Jianhuang Lai, and Yew-Soon Ong. 2024a. Generalizable and discriminative representations for adversarially robust few-shot learning. *IEEE Transactions on Neural Networks and Learning Systems*.

Junhao Dong, Melvin Wong, Sihan Xia, and Joel Wei En Tay. 2024b. Towards adversarially robust data-efficient learning with generated data. In *2024 IEEE Conference on Artificial Intelligence (CAI)*, pages 1422–1424. IEEE.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Wei Shen, Limao Xiong, Yuhao Zhou, Xiao Wang, Zhiheng Xi, Xiaoran Fan, Shiliang Pu, Jiang Zhu, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. 2024. LoRAMoE: Alleviating world knowledge forgetting in large language models via MoE-style plugin. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1932–1945, Bangkok, Thailand. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Benjamin Eysenbach, Vivek Myers, Ruslan Salakhutdinov, and Sergey Levine. 2024. Inference via interpolation: Contrastive representations provably enable planning and inference. In *Advances in Neural Information Processing Systems*, volume 37, pages 58901–58928. Curran Associates, Inc.

Yuchun Fan, Yongyu Mu, YiLin Wang, Lei Huang, Junhao Ruan, Bei Li, Tong Xiao, Shujian Huang, Xiaocheng Feng, and Jingbo Zhu. 2025. SLAM: Towards efficient multilingual reasoning via selective language alignment. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9499–9515, Abu Dhabi, UAE. Association for Computational Linguistics.

Han Fang, Zhifei Yang, Yuhan Wei, Xianghao Zang, Chao Ban, Zerun Feng, Zhongjiang He, Yongxiang Li, and Hao Sun. 2023. Alignment and generation adapter for efficient video-text understanding. In *ICCV (Workshops)*, pages 2783–2789.

Yujie Feng, Bo Liu, Xiaoyu Dong, Zexin Lu, Li-Ming Zhan, Xiao-Ming Wu, and Albert Lam. 2024. Continual dialogue state tracking via reason-of-select distillation. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 7075–7087.

Yujie Feng, Liming Zhan, Zexin Lu, Yongxin Xu, Xu Chu, Yasha Wang, Jiannong Cao, Philip S Yu, and Xiao-Ming Wu. 2025. Geoedit: Geometric knowledge editing for large language models. *arXiv preprint arXiv:2502.19953*.

Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.

Ziqi Gao, Qichao Wang, Aochuan Chen, Zijing Liu, Bingzhe Wu, Liang Chen, and Jia Li. 2024. Parameter-efficient fine-tuning with discrete Fourier transform. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 14884–14901. PMLR.

Han Guo, Philip Greengard, Eric Xing, and Yoon Kim. 2024. LQ-loRA: Low-rank plus quantized matrix decomposition for efficient language model finetuning. In *The Twelfth International Conference on Learning Representations*.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. Warp: Word-level adversarial reprogramming. *arXiv preprint arXiv:2101.00121*.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533, Doha, Qatar. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Wei Hu, Lechao Xiao, and Jeffrey Pennington. 2020. Provable benefit of orthogonal initialization in optimizing deep linear networks. In *International Conference on Learning Representations*.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. 2023. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5254–5276, Singapore. Association for Computational Linguistics.

Zhiyuan Hu, Yuliang Liu, Jinman Zhao, Suyuchen Wang, Yan Wang, Wei Shen, Qing Gu, Anh Tuan Luu, See-Kiong Ng, Zhiwei Jiang, and Bryan Hooi. 2024. Longrecipe: Recipe for efficient long context generalization in large language models. *Preprint*, arXiv:2409.00509.

Wei Huang, Weitao Du, and Richard Yi Da Xu. 2021. On the neural tangent kernel of deep networks with orthogonal initialization. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 2577–2583. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Gaurav Iyer, Gintare Karolina Dziugaite, and David Rolnick. 2024. Linear weight interpolation leads to transient performance gains. *Transactions on Machine Learning Research*.

Chen Jia. 2024. Adversarial moment-matching distillation of large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Yiyao Yu, Yuxiang Zhang, Dongdong Zhang, Xiao Liang, Hengyuan Zhang, Xingxing Zhang, Mahmoud Khademi, Hany Awadalla, Junjie Wang, Yujiu Yang, and Furu Wei. 2025. Chain-of-Reasoning: Towards Unified Mathematical Reasoning in Large Language Models via a Multi-Paradigm Perspective. *Preprint*, arXiv:2501.11110.

Mingyu Jin, Weidi Luo, Sitao Cheng, Xinyi Wang, Wenyue Hua, Ruixiang Tang, William Yang Wang, and Yongfeng Zhang. 2024. Disentangling memory and reasoning ability in large language models. *Preprint*, arXiv:2411.13504.

Minsoo Kim, Sihwa Lee, Wonyong Sung, and Jungwook Choi. 2024. RA-LoRA: Rank-adaptive parameter-efficient fine-tuning for accurate 2-bit quantized large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15773–15786, Bangkok, Thailand. Association for Computational Linguistics.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597.

Dawid J Kopiczko, Tijmen Blankevoort, and Yuki M Asano. 2023. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*.

Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Lei Li. 2024. Cpseg: Finer-grained image semantic segmentation via chain-of-thought language prompting. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 513–522.

Wenhao Li, Yuxin Zhang, Gen Luo, Daohai Yu, and Rongrong Ji. 2025a. Training long-context llms efficiently via chunk-wise optimization. *Preprint*, arXiv:2505.16710.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, Yingying Zhang, Fei Yin, Jiahua Dong, Zhiwei Li, Bao-Long Bi, Ling-Rui Mei, Junfeng Fang, Zhijiang Guo, Le Song, and Cheng-Lin Liu. 2025b. From system 1 to system 2: A survey of reasoning large language models. *Preprint*, arXiv:2502.17419.

Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2020. Exploring versatile generative language model via parameter-efficient transfer learning. *arXiv preprint arXiv:2004.03829*.

Wanlong Liu, Junying Chen, Ke Ji, Li Zhou, Wenyu Chen, and Benyou Wang. 2024a. Rag-instruct: Boosting llms with diverse retrieval-augmented instructions. *Preprint*, arXiv:2501.00353.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2024b. Gpt understands, too. *AI Open*, 5:208–215.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364.

Zeyu Liu, Souvik Kundu, Anni Li, Junrui Wan, Lianghao Jiang, and Peter Anthony Beerel. 2024c. Aflora: Adaptive freezing of low rank adaptation in parameter efficient fine-tuning of large models. *arXiv preprint arXiv:2403.13269*.

Yingfeng Luo, Tong Zheng, Yongyu Mu, Bei Li, Qinghong Zhang, Yongqi Gao, Ziqiang Xu, Peinan Feng, Xiaoqian Liu, Tong Xiao, et al. 2025. Beyond decoder-only: Large language models can be good encoders for machine translation. *arXiv preprint arXiv:2503.06594*.

Xuetao Ma, Wenbin Jiang, and Hua Huang. 2025. Problem-solving logic guided curriculum in-context learning for llms complex reasoning. *Preprint*, arXiv:2502.15401.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2023. A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*.

Maria-Elena Nilsback and Andrew Zisserman. 2008. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*.

Jingcheng Niu, Andrew Liu, Zining Zhu, and Gerald Penn. 2024. What does the knowledge neuron thesis have to do with knowledge? In *The Twelfth International Conference on Learning Representations*.

Jingcheng Niu, Wenjie Lu, and Gerald Penn. 2022. Does BERT rediscover a classical NLP pipeline? In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3143–3153, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. *arXiv preprint arXiv:1706.09254*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Boci Peng, Yongchao Liu, Xiaohe Bo, Sheng Tian, Baokun Wang, Chuntao Hong, and Yan Zhang. 2024a. Subgraph retrieval enhanced by graph-text alignment for commonsense question answering. In *Machine Learning and Knowledge Discovery in Databases. Research Track - European Conference, ECML PKDD 2024, Vilnius, Lithuania, September 9-13, 2024, Proceedings, Part VI*, pages 39–56.

Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. 2024b. Graph retrieval-augmented generation: A survey. *Preprint*, arXiv:2408.08921.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2020. Adapterfusion: Non-destructive task composition for transfer learning. *arXiv preprint arXiv:2005.00247*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Vivek Ramanujan, Mitchell Wortsman, Aniruddha Kembhavi, Ali Farhadi, and Mohammad Rastegari. 2020. What's hidden in a randomly weighted neural network? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11893–11902.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. *Advances in neural information processing systems*, 30.

Pengjie Ren, Chengshun Shi, Shiguang Wu, Mengqi Zhang, Zhaochun Ren, Maarten de Rijke, Zhumin Chen, and Jiahuan Pei. 2024. MELoRA: Mini-ensemble low-rank adapters for parameter-efficient fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3052–3064, Bangkok, Thailand. Association for Computational Linguistics.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752, Lisbon, Portugal. Association for Computational Linguistics.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2020. Adapterdrop: On the efficiency of adapters in transformers. *arXiv preprint arXiv:2010.11918*.

Ying Shen, Zhiyang Xu, Qifan Wang, Yu Cheng, Wenpeng Yin, and Lifu Huang. 2024. Multimodal instruction tuning with conditional mixture of LoRA. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 637–648, Bangkok, Thailand. Association for Computational Linguistics.

Shuhua Shi, Shaohan Huang, Minghui Song, Zhoujun Li, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, and Qi Zhang. 2024. ResLoRA: Identity residual mapping in low-rank adaption. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8870–8884, Bangkok, Thailand. Association for Computational Linguistics.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2024. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*.

Zexu Sun, Yiju Guo, Yankai Lin, Xu Chen, Qi Qi, Xing Tang, xiuqiang He, and Ji-Rong Wen. 2025. Uncertainty and influence aware reward model refinement for reinforcement learning from human feedback. In *The Thirteenth International Conference on Learning Representations*.

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Cheng zhong Xu. 2024. HydraloRA: An asymmetric loRA architecture for efficient fine-tuning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. 2022. Dylora: Parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. *arXiv preprint arXiv:2210.07558*.

Alex Wang. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Ruiyu Wang, Yu Yuan, Shizhao Sun, and Jiang Bian. 2025. Text-to-cad generation through infusing visual feedback in large language models. *Preprint*, arXiv:2501.19054.

Tiannan Wang, Wangchunshu Zhou, Yan Zeng, and Xinsong Zhang. 2023. EfficientVLM: Fast and accurate vision-language models via knowledge distillation and modal-adaptive pruning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13899–13913, Toronto, Canada. Association for Computational Linguistics.

Yaqing Wang, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. 2022. Adamix: Mixture-of-adapter for parameter-efficient tuning of large language models. *arXiv preprint arXiv:2205.12410*, 1(2):4.

Hui Wu, Yuanben Zhang, Zhonghe Han, Yingyan Hou, Lei Wang, Siye Liu, Qihang Gong, and Yunping Ge. 2025a. Cot-driven framework for short text classification: Enhancing and transferring capabilities from large to smaller model. *Knowledge-Based Systems*, page 113057.

Jie Wu, Haoling Li, Xin Zhang, Jianwen Luo, Yangyu Huang, Ruihang Chu, Yujiu Yang, and Scarlett Li. 2025b. Iterpref: Focal preference learning for code generation via iterative debugging. *Preprint*, arXiv:2503.02783.

Muling Wu, Wenhao Liu, Xiaohua Wang, Tianlong Li, Changze Lv, Zixuan Ling, Jianhao Zhu, Cenyuan Zhang, Xiaoqing Zheng, and Xuanjing Huang. 2024. Advancing parameter efficiency in fine-tuning via representation editing. *arXiv preprint arXiv:2402.15179*.

Haotian Xu, Xing Wu, Weinong Wang, Zhongzhi Li, Da Zheng, Boyuan Chen, Yi Hu, Shijia Kang, Jiaming Ji, Yingying Zhang, Zhijiang Guo, Yaodong Yang, Muhan Zhang, and Debing Zhang. 2025. Redstar: Does scaling long-cot data unlock better slow-reasoning systems? *Preprint*, arXiv:2501.11284.

Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhengsu Chen, XIAOPENG ZHANG, and Qi Tian. 2024. QA-loRA: Quantization-aware low-rank adaptation of large language models. In *The Twelfth International Conference on Learning Representations*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024a. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Hao Yang, Qianghua Zhao, and Lei Li. 2024b. Chain-of-thought in large language models: Decoding, projection, and activation. *Preprint*, arXiv:2412.03944.

Lei Yu, Jingcheng Niu, Zining Zhu, and Gerald Penn. 2024. Functional faithfulness in the wild: Circuit discovery with differentiable computation graph pruning. *Preprint*, arXiv:2407.03779.

Sheheryar Zaidi, Tudor Berariu, Hyunjik Kim, Jorg Bornschein, Claudia Clopath, Yee Whye Teh, and Razvan Pascanu. 2023. When does re-initialization work? In *Proceedings on "I Can't Believe It's Not Better! - Understanding Deep Learning Through Empirical Falsification" at NeurIPS 2022 Workshops*, volume 187 of *Proceedings of Machine Learning Research*, pages 12–26. PMLR.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2021. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*.

Jingfan Zhang, Yi Zhao, Dan Chen, Xing Tian, Huanran Zheng, and Wei Zhu. 2024. MiLoRA: Efficient mixture of low-rank adaptation for large language models fine-tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 17071–17084, Miami, Florida, USA. Association for Computational Linguistics.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023a. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*.

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Wenyue Hua, Haolun Wu, Zhihan Guo, Yufei Wang, Niklas Muennighoff, Irwin King, Xue Liu, and Chen Ma. 2025. A survey on test-time scaling in large language models: What, how, where, and how well? *Preprint*, arXiv:2503.24235.

Zhen-Ru Zhang, Chuanqi Tan, Haiyang Xu, Chengyu Wang, Jun Huang, and Songfang Huang. 2023b. Towards adaptive prefix tuning for parameter-efficient language model fine-tuning. *arXiv preprint arXiv:2305.15212*.

Tong Zheng, Yan Wen, Huiwen Bao, Junfeng Guo, and Heng Huang. 2025. Asymmetric conflict and synergy in post-training for llm-based multilingual machine translation. *arXiv preprint arXiv:2502.11223*.

Wangchunshu Zhou, Ronan Le Bras, and Yejin Choi. 2023. Modular transformers: Compressing transformers into modularized layers for flexible efficient inference. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10452–10465, Toronto, Canada. Association for Computational Linguistics.

A Hyperparameters

We record hyperparameters below for reproducibility: Table 6 for GLUE benchmark, Table 7 for E2E dataset, Table 8 for instruction turning, and Table 9 for image classification using ViT.

A.1 Tuning Strategy

Hyperparameter tuning directly relates to the fine-tuning performance, and it can be tricky. A good tuning strategy often saves users time during deployment. The hyperparameter tuning process for UORA is relatively straightforward as follows.

Rank r We find rank 16 is most likely sufficient for smaller models like RoBERTa and GPT-2. If computation resource allows, rank 32 often requires less tuning effort, and provides competitive performance out of box.

For large language models like llama and llama3, a minimum of rank 32 is needed. We observe that a rank < 32 could lead to zero interpolation occurrences. Higher ranks (e.g., r > 256) could cause overparameterization and instability.

Learning rate lr UORA benefits from a learning rate similar to VeRA's, typically higher than LoRA's. Given the small amount of trainable parameters, we find that 4e-2 is usually a good starting point.

Interpolation Factor α Alpha = 0.7 provides a balanced trade-off. A lower alpha < 0.5 introduces more noticeable turbulence in the training process (as the learned scaling vectors takes longer to adapt to the largely varied frozen matrices).

Threshold τ **and Count** k Threshold τ and count k together determine the number of reinitialization occurrences at a given index. The goal is to control the number of reinitializations within an optimal range. Too few reinitializations leads to similar performance as VeRA with a small rank; too many reinitializations may disrupt the orthogonality of frozen matrices and training stability.

We start with $k = 1$ and tune τ first. We adjust the Tau value to avoid overly frequent reinitializations. If tuning τ alone is insufficient to balance the number of reinitialized indices and total reinitializations, we then consider $k >= 2$, especially under noisier training regimes.

Model	Hyperparameter	SST-2	MRPC	CoLA	QNLI	RTE	STS-B
	Optimizer	AdamW					
	Warmup Ratio	0.06					
	LR Schedule	Linear					
	Initialization	Orthogonal uniform					
	Initial Value of d	0.1					
	Interpolation α	0.7					
	Count k	1					
	Seed	(0 42 88)					
RoBERTa base	# GPUs	1					
	Rank $r_q = r_v$	16	16	16	16	16	16
	Threshold τ	1e-5	1e-5	1e-5	1e-5	8e-6	8e-6
	Epochs	60	60	80	25	160	40
	Learning Rate (Head)	4E-3	1E-2	1E-2	4E-3	1E-2	1E-2
	Learning Rate (UORA)	4E-3	1E-2	1E-2	1E-2	4E-3	1E-2
	Max Seq. Len.	512					
	Batch Size	32					
RoBERTa large	# GPUs	4					
	Rank $r_q = r_v$	32	32	32	32	32	32
	Threshold τ	1e-5	1e-5	1e-5	8e-6	8e-6	8e-6
	Epochs	10	40	40	20	40	20
	Learning Rate (Head)	4E-3	1E-2	6E-3	2E-4	2E-3	5E-3
	Learning Rate (UORA)	1E-2	1E-2	1E-2	1E-2	2E-2	1E-2
	Max Seq. Len.	128					
	Batch Size Per GPU	32					

Table 6: Hyperparameters for GLUE benchmark.

Model	Hyperparameter	LoRA	VeRA	RED	UORA
	Optimizer	AdamW			
	Warmup Steps	500			
	Epochs	5			
	Label Smooth	0.0			
	Batch Size	8			
	LR Schedule	Linear			
	Seed	(42 43 44)			
	# GPUs	1			
	Initial Value of d	-	0.1	-	0.1
	Initialization	-	Kaiming uniform	-	Orthogonal uniform
	Threshold	-	-	-	1
GPT-2 Medium	Rank r	8	1024	-	32
	Learning Rate	6E-2	2E-2	6E-2	1E-2
GPT-2 Large	Rank r	8	1024	-	32
	Learning Rate	4E-3	6E-3	2E-2	2E-2

Table 7: Hyperparameters for E2E benchmark.

	LoRA	VeRA	UORA
#GPU		1	
Optimizer		AdamW	
Warmup Ratio		0.03	
Batch Size		4	
Gradient Accumulation Steps		4	
Epochs		3	
LR Scheduler		Cosine	
Weight Decay		0.0	
Cutoff Length		256	
Rank r	8	1024	32
Learning Rate	3e-4	4e-2	4e-2

Table 8: Hyperparameter setup for Instruction Tuning. LoRA alpha = 16. UORA τ = 5e-5; k = 1.

Model	**Hyperparameter**	**CIFAR100**	**Food101**	**Flowers102**	**RESISC45**
	Weight Decay		0.0		
	Optimizer		AdamW		
	LR Schedule		Linear		
	Rank r		32		
	epoch		10		
	Seed		(42,43,44)		
Base	LR-head (Head only)	5e-5	5e-5	5e-5	5e-5
	LR (Full)	5e-5	5e-5	5e-5	5e-5
	LR-head (LoRA)	4e-3	4e-3	4e-3	4e-3
	LR (LoRA)	4e-3	4e-3	4e-3	4e-3
	LR-head (VeRA)	4e-3	4e-3	4e-3	4e-3
	LR (VeRA)	4e-2	4e-2	4e-2	4e-2
	LR-head (UORA)	4e-3	4e-3	4e-3	4e-3
	LR (UORA)	4e-2	4e-2	4e-2	4e-2
Large	LR-head (Head only)	5e-5	5e-5	5e-5	5e-5
	LR (Full)	5e-5	5e-5	5e-5	5e-5
	LR-head (LoRA)	4e-3	4e-3	4e-3	4e-3
	LR (LoRA)	4e-3	4e-3	4e-3	4e-3
	LR-head (VeRA)	4e-3	4e-3	4e-3	4e-3
	LR (VeRA)	4e-2	4e-2	4e-2	4e-2
	LR-head (UORA)	4e-3	4e-3	4e-3	4e-3
	LR (UORA)	4e-2	4e-2	4e-2	4e-2

Table 9: Hyperparameter setup for ViT on the image classification benchmarks.

B Ablation Study

This section depicts the initialization comparison and sensitivity analysis of UORA-specific hyperparameters (τ, k, α).

Threshold τ. τ controls the magnitude threshold to trigger reinitialization. If a dimension in $\vec{d}$ becomes lower than τ, it is considered as less efficient in modeling the weight distribution of target layers. We conducted a focused experiment on LLaMA 7B on the AddSub dataset to evaluate the impact of the threshold τ.

As shown in Table 11, a higher value of τ leads to more frequent reinitialzziation; excessive occurrences could introduce performance degradation. τ = 1e-4 provides the best performance in this ex-

Initialization	MRPC
Xavier Uniform	88.72
Orthogonal Uniform	88.72
Kaiming Uniform	87.99
Random Uniform	72.05

Table 10: Impact of Matrix Initialization

periment. As τ decreases further, the performance begins to drop, which highlights the importance of a balanced reinitialization frequency.

Threshold τ	AddSub
5e-4	0.6759
1e-4	0.8095
5e-5	0.7620
1e-5	0.7063

Table 11: Impact of Threshold τ

Count k. Count k controls the number of training steps where a dimension is below τ before reinitialization triggers. A higher k reduces the frequency of reinitialization and makes UORA conservative about the current frozen matrices. Special values like $k = 0$ disables UORA reinitialization mechanism; another case is that when k is large enough that the consecutive count is never reached. We conduct empirical study to make sense of the appropriate range for k. We observe that given a tuned threshold τ, $k \in \{1, 2, 3\}$ shows promising performance.

Count k	MRPC
0	89.95
1	90.44
2	89.99
3	89.95
4	89.95

Table 12: Impact of Count k

Interpolation Factor α. α controls the magnitude of change when modifying a frozen matrix during reinitialization. A higher α leads to more stabilized training process at a cost of inefficient parameter modeling. When $\alpha = 1$, UORA can be viewed a close variant of VeRA where the initialization method adopts orthogonal uniform. Table 13 shows the impact of interpolation factor α on the E2E dataset. We observe that $\alpha = 0.7$ provides the best trade-off between performance and stability as it preserves a better approximate orthogonality than smaller α's.

α	0.3	0.5	0.7	1
BLEU	63.90	63.75	66.67	65.78
NIST	8.35	8.25	8.61	8.49
METEOR	44.96	45.20	45.37	45.58
ROUGE_L	67.64	68.51	68.82	69.01
CIDEr	2.30	2.32	2.35	2.32

Table 13: Sensitivity analysis of interpolation factor α.

C Instruction Tuning Examples

Table 14 shows an example of the multi-turn questions in MT-Bench. Table 15 shows the LoRA answers and UORA for the same question.

Q1	Extract the following information from the presented texts: The name of the book, the author, the main character, the year of publication. Output in the format of "main character, book, author, year of publication", one book per line. a) In the realm of wizarding literature, a true standout is the work of J.K. Rowling. One of her books that left an indelible mark is 'Harry Potter and the Philosopher's Stone'. This iconic tale, published in 1997, tells the story of Harry, a young orphan who discovers his magical abilities on his 11th birthday. Soon, he finds himself at the Hogwarts School of Witchcraft and Wizardry, a place teeming with magic and adventure, located somewhere in Scotland. b) The magic of Middle-earth has entranced readers worldwide, thanks to the brilliance of J.R.R. Tolkien. In one of his seminal works, 'The Lord of the Rings: The Fellowship of the Ring', published in 1954, we meet Frodo Baggins, a brave hobbit tasked with the perilous quest of destroying the One Ring. The epic journey takes him from the peaceful Shire to the tumultuous regions of Middle-earth. c) In a galaxy far, far away, the imagination of L.E. Starlighter gives us 'The Prism Galaxy Chronicles: The Awakening of the Starcaster'. Published in 2028, the story is about Zylo, a humble spaceship mechanic, who unexpectedly discovers he's a Starcaster - a rare individual with the power to manipulate stardust. Set against the backdrop of an interstellar empire in turmoil, Zylo's destiny unfolds on numerous alien worlds, each with its unique cosmic charm.
Q2	Reformulate your earlier reply, output it in JSON format and only include books published after 1980.

Table 14: No. 131 question in MT-Bench

	LoRA	UORA
Q1	The name of the book is ’Harry Potter and the Philosopher’s Stone’, the author is J.K. Rowling, the main character is Harry Potter, and the year of publication is 1997.	Here are the answers to your question: a) The main character is Harry Potter, the book is ’Harry Potter and the Philosopher’s Stone’, the author is J.K. Rowling, and the year of publication is 1997. b) The main character is Frodo Baggins, the book is ’The Lord of the Rings: The Fellowship of the Ring’, the author is J.R.R. Tolkien, and the year of publication is 1954. c) The main character is Zylo, the book is ’The Prism Galaxy Chronicles: The Awakening of the Starcaster’, the author is L.E. Starlighter, and the year of publication is 2028.

Q2	{ "books": [
	{
	"name": "Harry Potter and the Philosopher’s Stone",
	"author": "J.K. Rowling",
	"main_character": "Harry Potter",
	"year_of_publication": 1997
	},
	{
	"name": "The Lord of the Rings: The Fellowship of the Ring",
	"author": "J.R.R. Tolkien",
	"main_character": "Frodo Baggins",
	"year_of_publication": 1954
	},
	{
	"name": "The Prism Galaxy Chronicles: The Awakening of the Starcaster",
	"author": "L.E. Starlighter",
	"main_character": "Zylo",
	"year_of_publication": 2028
	}
	] }
		Here is the reformulated answer:
		a) { "main character": "Harry Potter", "book": "Harry Potter and the Philosopher’s Stone", "author": "J.K. Rowling", "year of publication": "1997" }
		b) { "main character": "Frodo Baggins", "book": "The Lord of the Rings: The Fellowship of the Ring", "author": "J.R.R. Tolkien", "year of publication": "1954" }
		c) { "main character": "Zylo", "book": "The Prism Galaxy Chronicles: The Awakening of the Starcaster", "author": "L.E. Starlighter", "year of publication": "2028" }

Table 15: Example answers to MT-Bench questions. Answers formatted according to newline character in the model output.