Grow-on-Demand: Sparse and Adaptive Expert Expansion for Continual Instruction Tuning

Ying Zhang[1], **Xingyue Guo**[1*], **Yu Zhao**[1], **Xuhui Sui**[1],
Baohang Zhou[2], **Xinying Qian**[1], **Xiaojie Yuan**[1]

[1]College of Computer Science, VCIP, DISSec Center, Nankai University, Tianjin, China
[2]School of Software, Tiangong University, Tianjin, China
yingzhang@nankai.edu.cn, guoxingyue@dbis.nankai.edu.cn, zhaoyu@dbis.nankai.edu.cn, suixuhui@dbis.nankai.edu.cn,
zhoubaohang@tiangong.edu.cn, qianxinying@dbis.nankai.edu.cn, yuanxj@nankai.edu.cn

Abstract

Continual instruction tuning aims to incrementally adapt large language models to new tasks without forgetting previously acquired knowledge. Existing approaches often struggle to balance plasticity and stability. Replay-based methods retrain on historical data, which raises privacy concerns. Architecture-based methods allocate task-specific components, resulting in significant parameter growth. To address this, we consider a structure-sharing strategy that enables parameter reuse across similar tasks and expands only when necessary, avoiding any data replay. Specifically, we introduce Grow-on-Demand (GoD-MoE), a parameter-efficient framework that is based on sparse and adaptive expert module expansion for continual instruction tuning. GoD-MoE inserts multiple LoRA-based experts into attention layers and dynamically activates a small subset of experts for each task. To avoid redundant parameter growth, we develop an Expert Demand Detector that determines whether new experts are added, facilitating adaptive structural sharing and minimizing parameter overhead. We conduct comprehensive experiments on the TRACE benchmark, demonstrating that GoD-MoE achieves state-of-the-art performance. Furthermore, it effectively mitigates catastrophic forgetting and even outperforms several advanced replay-based baselines.

Code — https://xyguo1229.github.io/GoD-MoE/

Introduction

In recent years, large language models (LLMs) (Achiam et al. 2023; Touvron et al. 2023; Team et al. 2024) have achieved significant progress in natural language processing tasks, demonstrating strong capabilities in language understanding and generation. Instruction tuning (Wang et al. 2022a) improves generalization and alignment, enabling models to follow diverse instructions. However, as application scenarios expand, models tend to forget previously acquired knowledge and instructions during continual instruction tuning (Shi et al. 2024; Zheng et al. 2025). This phenomenon is known as catastrophic forgetting (McCloskey and Cohen 1989). Consequently, continual instruction tuning must effectively balance plasticity (the ability to acquire new knowledge) and stability (the ability to preserve previous knowledge) (Mermillod, Bugaiska, and Bonin 2013) to mitigate catastrophic forgetting of existing knowledge while adapting to new tasks and instructions.

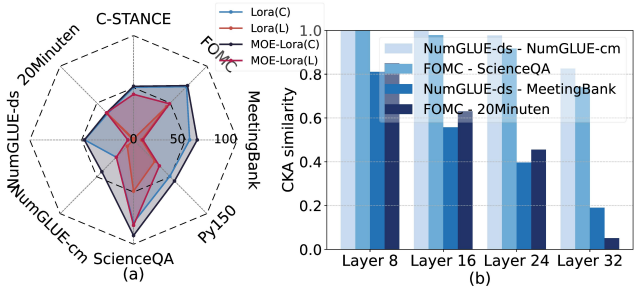

Figure 1: (a) Performance between standard LoRA and MoE-LoRA. Suffix (C) indicates performance on the current task during sequential tuning, and suffix (L) indicates the final performance of each task after continual tuning up to the last task. (b) Detailed CKA similarity comparison across different task pairs and model layers.

Existing continual learning methods generally fall into three categories: replay-based methods (Rolnick et al. 2019; Han et al. 2020) that retrain on historical data, regularization-based methods (Jin et al. 2021; Wang et al. 2023a)that impose constraints to preserve past knowledge, and architecture-based methods (Guo et al. 2024; Qin, Chen, and Joty 2023) that introduce additional structures for each task. Recent advanced continual instruction tuning methods of LLMs often combine these strategies with lightweight adapter modules such as LoRA (Hu et al. 2022). However, methods relying on data replay (Jung and Kim 2024; Ren et al. 2024) raise privacy concerns and are often impractical in real-world scenarios. In addition, methods that assign separate architectures for each task (Guo et al. 2025; Wang et al. 2023a) lead to progressive parameter growth as tasks accumulate, significantly reducing training efficiency.

To address these challenges, we conduct the following analysis. We begin with a preliminary analysis of standard LoRA fine-tuning (Hu et al. 2022) and observe that it suffers from severe catastrophic forgetting in continual instruction tuning as shown in Figure 1(a). To better adapt to di-

*Coresponding author.

verse tasks, we introduce a mixture-of-experts (MoE) (Jacobs et al. 1991) structure by dividing the low-rank LoRA matrices into multiple experts (Liu et al. 2023), allowing each task to select a subset of experts to access specialized knowledge. Multi-expert LoRA tuning improves performance on the current task and enhances model plasticity. However, updating a separate set of parameters for each new task often leads to overwriting previously acquired knowledge, resulting in persistent forgetting. Furthermore, training a dedicated MoE-LoRA structure for every task introduces substantial parameter overhead and increases training cost.

As a further observation, intuitively, even tasks from different domains may share structural or semantic similarities. Motivated by this, we further analyze representation similarity using Centered Kernel Alignment (CKA) (Kornblith et al. 2019) across tasks. As shown in Figure 1(b), tasks such as *NumGLUE-ds* and *NumGLUE-cm*, or *FOMC* and *ScienceQA*, are highly similar, while *NumGLUE-ds* and *MeetingBank*, or *FOMC* and *20Minuten*, show low similarity. These observations motivate us to explore a structure-sharing strategy: reusing existing experts for similar tasks and expanding new experts only when demanded. This strategy aims to preserve previously acquired knowledge while adapting to new tasks, enabling greater stability in continual instruction tuning and avoiding any data replay.

Based on these observations, we propose Grow-on-Demand (GoD-MoE), a sparse and adaptive expert expansion framework for continual instruction tuning. Specifically, GoD-MoE introduces a mixture-of-experts structure into the attention layers by decomposing the variable parts of the weights into multiple LoRA-based expert matrices, enabling dynamic expert routing and activation. After training on each task, a subset of active experts is frozen to preserve acquired knowledge. When a new task arrives, GoD-MoE samples a small set of its training data and employs a lightweight Expert Demand Detector to assess whether to add additional experts. New experts are added only if the existing frozen experts fail to capture the characteristics of the current task sufficiently. As the number of tasks grows, existing experts can be reused or combined to cover different task types and instructions, reducing the demand for continual expert growth for every task. Sparse and adaptive expert expansion effectively prevents parameter explosion, allowing continual expansion while efficiently utilizing the parameter space and controlling computational costs.

Our contributions can be summarized as follows:

- We propose GoD-MoE, a parameter-efficient framework for continual instruction tuning without any data replay, which leverages sparse and adaptive expert expansion to balance plasticity and scalability across tasks.
- We introduce an Expert Demand Detector that determines whether new experts are demanded, effectively reducing redundant parameter growth.
- We conduct extensive experiments on the TRACE benchmark and achieve state-of-the-art performance. GoD-MoE also outperforms several replay-based methods in mitigating forgetting.

Related Work

Continual Learning for LLM

Continual Learning (CL) (Wang et al. 2024) enables models to learn sequentially from multiple tasks while retaining prior knowledge and mitigating catastrophic forgetting. Existing methods are typically categorized into three groups: replay-based, regularization-based, and architecture-based methods. *Replay-based methods* store and retrain past data to mitigate forgetting, which is one of the most direct and effective strategies. Some studies (Rolnick et al. 2019; Han et al. 2020) have enhanced replay-based strategies by introducing techniques such as dynamic sampling of informative examples and generating data using LLMs. *Regularization-based methods* (Wang et al. 2023a; Ren et al. 2024) constrain model updates to prevent drastic changes to important parameters. Common techniques include adding regularization terms (Wang et al. 2023a) or distilling predictions from previous tasks into the current model (Ren et al. 2024). *Architecture-based methods* add new modules to handle new tasks while preserving parameters important for previous ones. These approaches include adapter-based (Qin, Chen, and Joty 2023; Ke et al. 2023) and prompt-based (Wang et al. 2022b; Razdaibiedina et al. 2023) methods. Adapter-based methods insert lightweight trainable adapters into the model, assigning task-specific ones for new tasks to ensure parameter isolation. Prompt-based methods guide model behavior through task-specific prompts, enabling adaptation to new tasks without altering core model parameters. Considering privacy and parameter efficiency, our work develops a continual learning framework that avoids data replay and prevents excessive architectural overhead as tasks increase.

Mixture of Experts

Mixture-of-Experts (MoE) enables high model capacity at low computational cost by activating only a subset of expert networks for each input. Recent studies have widely explored MoE architectures for scalable and effective continual learning (Mu and Lin 2025). Early efforts (Lee et al. 2020) integrate MoE with continual learning by expanding and selecting experts through generative or input-driven gating. Recent studies extend the MoE framework by incorporating LoRA-based experts for parameter-efficient fine-tuning (PEFT) (Ding et al. 2023) of LLMs. MoLA (Gao et al. 2024) investigates flexible layer-wise expert allocation strategies, PMoE (Jung and Kim 2024) incorporates expert modules only in the deeper layers of the transformer layers, and AlphaLoRA (Qing et al. 2024) explores expert placement across layers guided by HT-SR theory. D-MoLE (Ge et al. 2025) further expands the MoE framework to handle multimodal instruction tuning of LLMs. Unlike the above approaches that allocate task-specific experts, we promote expert sharing across tasks and add new experts only when the existing ones fail to meet the demands of a new task.

Method

Continual Learning Setup

Continual learning aims to develop models that can learn from a sequence of tasks without forgetting previously ac-

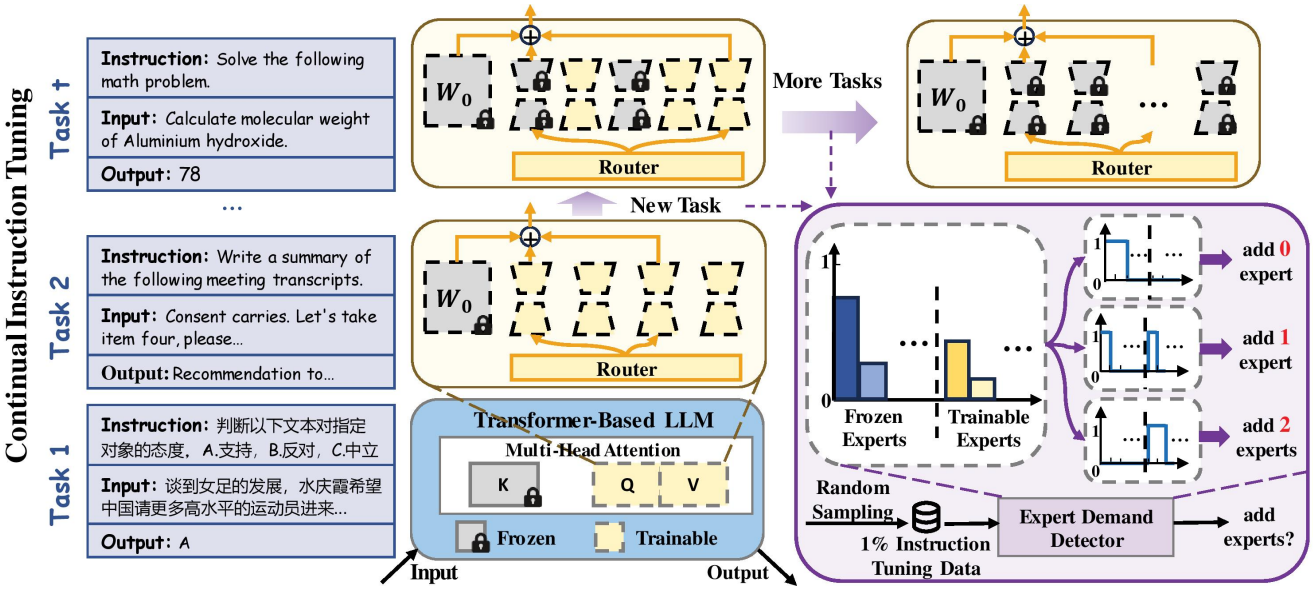

Figure 2: Overall framework of GoD-MoE. As new tasks arrive, the model decomposes the variant parts of the attention weights (Q, V) into multiple LoRA-based experts. After each task, the most activated experts are frozen to retain knowledge. A lightweight Expert Demand Detector samples 1% of the new task data to decide whether to expand experts. New experts are added only when demanded, allowing expert reuse across tasks and preventing parameter explosion.

quired knowledge. A sequence of tasks $\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_T\}$ arrives in order, where each task $\mathcal{D}_t$ is introduced to the model individually. Each task consists of a set of labeled examples $\mathcal{D}_t = \{(x_i^t, y_i^t)\}_{i=1}^{n_t}$, where $x_i^t \in \mathcal{X}_t$ denotes the input and $y_i^t \in \mathcal{Y}_t$ denotes the corresponding label. At any time, the model only has access to the current task $\mathcal{D}_t$.

At each time step t, the model learns from the current task $\mathcal{D}_t$ by updating its parameters, while preserving performance on previous tasks. Given a prediction model with parameters Θ, the goal of continual learning is to optimize the following objective over the task sequence:

$$\max_{\Theta} \sum_{t=1}^{T} \sum_{(x,y)\in\mathcal{D}_t} \log p_\Theta(y \mid x) \tag{1}$$

A core challenge in continual learning is to adapt to non-stationary data distributions without catastrophic forgetting, which requires maintaining performance on previously learned tasks when learning new ones.

Overview

In this paper, we propose GoD-MoE, a parameter-efficient continual instruction tuning framework for sparse and adaptive expert expansion, enhancing LLMs' ability to learn diverse tasks continually. As illustrated in Figure 2, GoD-MoE introduces a mixture-of-experts mechanism into the attention layers by decomposing the variant part of the weights Q and V into multiple LoRA-based expert matrices. This enables dynamic expert routing and activation (see the *Sparse and Adaptive Expert Expansion* section for details).

After training on each task, the most two active experts are frozen to retain previous knowledge. When a new task arrives, we first randomly sample 1% of its training data and employ a lightweight *Expert Demand Detector* to determine whether the current expert set is sufficient for the new task. Based on this assessment, GoD-MoE selectively expands its new experts as needed. (see the *Expert Expansion Detector* section for details).

As the number of tasks increases, different experts tend to capture knowledge representations related to specific task types or instructional intents. The diverse combinations of existing experts can effectively support a wide range of task scenarios, allowing new tasks to be handled without adding more experts. This maintains a sublinear growth in trainable parameters as tasks increase, enabling continual expansion and efficient utilization of the parameter space under a controlled computational cost.

Sparse and Adaptive Expert Expansion

Low-Rank Adaptation (LoRA) (Hu et al. 2022) is widely used for parameter-efficient fine-tuning of LLMs due to its efficiency and effectiveness. However, standard LoRA inserts a fixed low-rank branch into each attention weight matrix, which limits its expressiveness for complex tasks. To enhance task adaptability and promote parameter sharing across tasks, inspired by (Liu et al. 2023) and (Dou et al. 2023), we extend LoRA with a Mixture-of-Experts (MoE) structure. Specifically, the fixed LoRA branch is replaced with a pool of LoRA-based experts. During each forward pass, only a small subset of experts is activated, effectively increasing model capacity without extra computation cost.

In the continual instruction tuning setting, we update a set of LoRA matrices for the Q and V weight in each attention layer during every training phase. Specifically, the feedfor-

ward output of each layer is defined as:

$$h = W_0 x + \sum_{i \in \text{Top}_2(x)} G_i(x) \cdot B_i A_i x \tag{2}$$

where W_0 denotes the original pretrained weight matrix, A_i and B_i are the LoRA matrices of the i-th LoRA expert, and $G_i(x)$ is the activation weights generated by a gating function. The set $\text{Top}_2(x)$ represents the top-2 activated experts for input x. The gating weights are computed as follows:

$$G(x) = \text{Softmax}(\text{Top}_2(xW_g)) \tag{3}$$

where W_g is a learnable parameter matrix. After training on each task, we record the activation frequency of each expert and freeze the top-2 most frequently used experts to preserve knowledge of previous tasks. Specifically, for the first task, we freeze two experts. For each subsequent task, we allow the maximum number of frozen experts in each layer to increase by at most one, in order to balance plasticity-stability.

When introducing a new LoRA expert with the parameters A_{N+1} and B_{N+1}, the weights are initialized by averaging the parameters of the existing experts as $A_{N+1} = \frac{1}{N}\sum_{i=1}^{N} A_i$ and $B_{N+1} = \frac{1}{t}\sum_{i=1}^{N} B_i$, where N denotes the number of existing experts. And for the gating parameters, we expand the routing weight matrix $W_g \in \mathbb{R}^{N \times D}$ by appending an additional row. Specifically, we compute the mean of the existing expert routing vectors and add Gaussian noise $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$ to initialize the new expert's routing vector w_{new}. The new vector is concatenated to the original matrix to produce the updated routing weight matrix as:

$$W_g' = \text{Concat}(W_g, w_{\text{new}}) \in \mathbb{R}^{(N+1) \times D} \tag{4}$$

Expert Demand Detector

For each new task $\mathcal{D}_t$, we introduce an Expert Demand Detector to determine whether the current expert set is sufficient for the task and whether new experts need to be added. Specifically, we perform both forward and backward passes on 1% of the training samples from $\mathcal{D}_t$ with the current model parameters. We then compute the activation distributions for frozen and trainable experts separately and sort them in descending order of their probability, resulting in two probability vectors: P^{freeze} and $P^{\text{trainable}}$. These two vectors are concatenated to form the full distribution vector:

$$P = \left[P^{\text{freeze}} \parallel P^{\text{trainable}}\right] \in \mathbb{R}^N \tag{5}$$

where $\parallel$ denotes the concatenation operation.

Frozen experts refer to those that have been frequently used in previous tasks, while trainable experts are those that have been used less or not yet activated. To evaluate the compatibility between the current task and the existing expert set, we compare the activation distribution P with three prototypical expert usage patterns $\overline{P}$, each representing a different level of compatibility with the existing experts. These patterns $\overline{P}$ are manually constructed binary vectors and have the same dimensions as P:

- *Pattern A (High Match)*:

$$\overline{P}^{(A)} = [1, 1, 0, \dots] \parallel [0, 0, \dots] \in \mathbb{R}^N$$

  This indicates that the current task primarily relies on frozen experts, suggesting that the existing expert set is sufficient. **No new expert is added.**

- *Pattern B (Partial Match)*:

$$\overline{P}^{(B)} = [1, 0, \dots] \parallel [1, 0, \dots] \in \mathbb{R}^N$$

  This indicates that frozen experts are partially sufficient, but a few new experts are still needed to capture task-specific representations. **One new expert is added.**

- *Pattern C (Low Match)*:

$$\overline{P}^{(C)} = [0, 0, \dots] \parallel [1, 1, 0, \dots] \in \mathbb{R}^N$$

  This indicates that the task primarily relies on previously unused experts and needs to introduce new experts to enhance task representation. **Two new experts are added.**

To reduce instability in pattern selection, we apply Monte Carlo Dropout (Gal and Ghahramani 2016) at inference by performing M stochastic forward passes, resulting in a set of perturbed activation distributions $P_m \in \mathbb{R}^{N \times M}$. The similarity to prototypical patterns is computed as:

$$S = \overline{P}^{\top} P \in \mathbb{R}^{3 \times M}, \overline{P} = \left[\overline{P}^{(A)}, \overline{P}^{(B)}, \overline{P}^{(C)}\right] \in \mathbb{R}^{N \times 3} \tag{6}$$

For each prototype i, the similarity mean and variance are $\mu_i = \frac{1}{M}\sum_{j=1}^{M} S_j^{(i)}$, $\sigma_i^2 = \frac{1}{M}\sum_{j=1}^{M}\left(S_j^{(i)} - \mu_i\right)^2$. The final pattern is determined by the Upper Confidence Bound (UCB) criterion (Auer, Cesa-Bianchi, and Fischer 2002):

$$i^* = \arg\max_{i \in A,B,C} (\mu_i - \lambda \sigma_i) \tag{7}$$

where λ is a hyperparameter controlling uncertainty.

Training and Inference Protocol

For the first task, we initialize $N = 4$ LoRA experts for each layer. As new tasks arrive, experts are frozen or expanded according to the strategy mentioned above. During training, we obtain the reference vector $\mathbf{p}_t$ for each task $\mathcal{D}_t$ by averaging the final-layer representations of all input samples from the backbone. At inference time, for a given input x_{test}, we extract its final-layer representation $\mathbf{h}_{\text{test}}$ and compute cosine similarity between $\mathbf{h}_{\text{test}}$ and each stored reference vector: $\{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_T\}$. Finally, the routing strategy associated with the most similar reference $\mathbf{p}_t^*$ is selected for inference. This enables automatic expert selection without requiring access to the test task ID.

Experiments

Experimental Setup

Dataset and Evaluation Metrics We conduct experiments on the TRACE benchmark (Wang et al. 2023b), which consists of 8 tasks across diverse categories, covering domain-specific instructions, multilingual understanding, code completion, and mathematical reasoning. These tasks are challenging and novel enough that most LLMs have not been trained on them. Following (Wang et al. 2023b),

Method	*C-STANCE*	*FOMC*	*MeetingBank*	*Py150*	*ScienceQA*	*NumGLUE-cm*	*NumGLUE-ds*	*20Minuten*	Avg	BWT
Few-shot	40.0	48.3	19.8	52.2	62.8	28.4	20.3	39.5	38.9	-
Multi-task	48.3	71.8	57.0	56.8	89.9	46.9	60.6	36.3	58.4	-
EWC	24.5	37.7	16.8	7.7	37.7	6.2	3.1	23.5	19.7	-18.9
L2P	7.7	43.5	13.5	8.1	24.8	4.9	2.8	21.2	15.8	-15.7
PP	5.6	48.0	9.4	2.5	23.4	7.4	8.0	23.2	15.9	-16.1
O-LoRA	48.2	33.6	40.9	53.0	58.2	23.5	45.5	26.4	41.2	-6.2
I-LoRA	42.9	59.5	23.7	42.5	47.1	22.2	43.4	28.2	38.7	-7.0
PMoE	47.5	68.3	38.5	**55.1**	72.4	**41.6**	59.2	26.1	51.1	**+12.2**
HiDe-LLaVA	37.9	64.9	21.1	38.1	78.8	13.6	27.4	33.4	39.4	-11.3
GoD-MoE	**49.1**	**68.8**	**48.6**	54.8	**88.1**	38.3	**60.3**	**37.6**	**55.7**	-3.6
Full-FT	45.4	60.9	45.7	51.2	63.7	27.2	54.8	40.8	48.7	-8.3
LoRA	9.7	45.0	7.7	29.0	49.1	8.6	0.3	35.5	23.1	-25.4
LoRA-RE	48.1	69.6	43.4	53.3	70.1	38.7	50.5	25.9	50.0	+7.5
MoE-LoRA	43.7	49.0	9.0	35.2	81.9	23.5	5.5	36.8	35.6	-16.9

Table 1: Comparison with baselines on the TRACE benchmark (Last, Avg, BWT). *Few-shot* denotes 6-shot in-context learning with a frozen backbone and serves as a reference without fine-tuning. *Multi-task* trains on all tasks jointly and serves as an upper bound. The bottom block presents results from continual learning baselines and several direct tuning methods for reference. Compared with continual learning baselines, the best results are highlighted in **bold**, and the second best are underlined.

Method	MMLU	GSM	BBH	BoolQ	PIQA
LLaMA2-7b	46.6	26.1	40.2	70.6	76.2
O-LoRA	46.7	21.9	40.3	79.6	76.9
I-LoRA	23.9	24.7	29.1	74.9	67.3
PMoE	46.8	10.7	37.1	80.8	75.7
HiDe-LLaVA	47.6	18.9	34.4	74.2	76.9
GoD-MoE	48.3	22.1	39.0	77.9	77.4

Table 2: Evaluation results on general LLM benchmarks

each task contains 5,000 training samples, presented in the order: C-STANCE, FOMC, MeetingBank, Py150, ScienceQA, NumGLUE-cm, NumGLUE-ds, 20Minuten. Detailed dataset statistics are provided in Appendix.

To evaluate the model's general ability after continual learning, we evaluate the model on five well-known LLM benchmarks: MMLU (Hendrycks et al. 2020), GSM (Cobbe et al. 2021), BBH (Suzgun et al. 2022), BoolQ (Clark et al. 2019), and PIQA (Bisk et al. 2020).

Following standard metrics (Wang et al. 2023b) in continual learning, we report $Last_i$, *Avg*, and *backward transfer (BWT)* metrics to evaluate continual learning performance. Specifically, $Last_i$ denotes the evaluation score on the i-th task after completing the last task, *Avg* is the average of all $Last_i$ scores, *BWT* measures forgetting by comparing final and initial performance on each task, and is defined as:

$$BWT = \frac{1}{T}\sum_{i=1}^{T}(R_{T,i} - R_{i,i}) \tag{8}$$

where $R_{T,i}$ denotes the evaluation score on the i-th task after training on the last T-th task ($i \leq T$).

Baselines We compare our GoD-MoE with several representative baselines across different categories: (1) Few-shot (6-shot) in-context learning with a frozen backbone; (2) Multi-task fine-tuning, where all task training sets are jointly trained instead of sequential training; (3) regularization-based method EWC (Kirkpatrick et al. 2017); (4) prompt-based methods L2P (Wang et al. 2022b) and PP (Razdaibiedina et al. 2023); and (5) advanced continual instruction tuning methods of LLMs, including O-LoRA (Wang et al. 2023a), I-LoRA (Ren et al. 2024), PMoE (Jung and Kim 2024), and HiDe-LLaVA (Guo et al. 2025), where PMoE and I-LoRA adopt replay mechanisms.

Implementation Details For fair comparison, we adopt LLaMA2-7b (Touvron et al. 2023) as the backbone for GoD-MoE and all baselines. We set the batch size to 128 and the learning rate to 3e-4. LoRA is configured with rank 8 and scaling factor 16. The hyperparameter λ is fixed at 1. When no trainable experts are available, the corresponding similarity score $S^{(C)}$ is set to 0.5 by default. Each task is trained for 5 epochs. We initialize 4 LoRA experts for the first task, and freeze the two most frequently used experts after each task. All experiments are performed on NVIDIA RTX A6000 GPUs with 48 GB memory.

Overall Performance

Main Results Table 1 presents the overall performance of our method and baselines on the TRACE benchmark. The results show that God-MoE outperforms all baselines and effectively mitigates catastrophic forgetting during continual instruction tuning. Traditional continual learning methods (EWC, P2P, PP) perform poorly. Recent advanced continual instruction tuning methods for LLMs show improved performance over few-shot in-context learning, but still underperform compared to multi-task learning. In contrast, GoD-MoE achieves the best overall performance, even com-

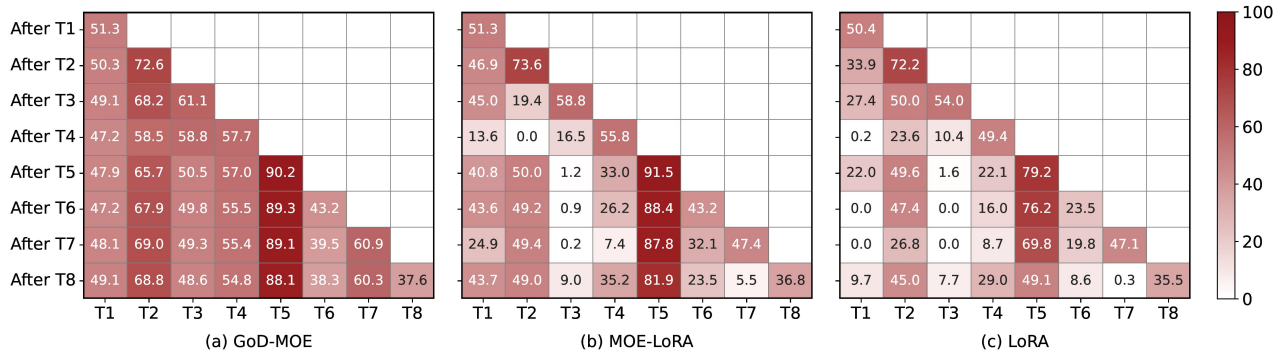

Figure 3: Comparison of knowledge Retaining performance. The values indicate the model's performance on previous tasks after training on each new task.

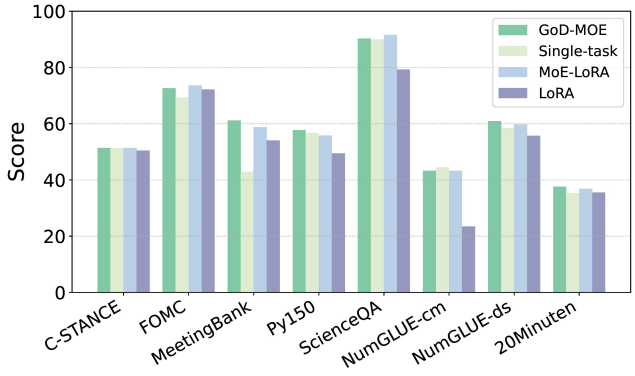

Figure 4: Comparison of knowledge acquisition performance. *Single-task* refers to training each task individually.

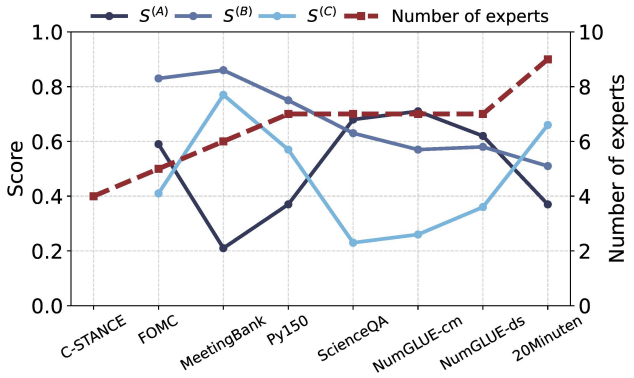

Figure 5: Analysis of the similarity score $S^{(i)}$ and the number of experts for each task.

pared to methods with replay. On the *ScienceQA*, it outperforms the second-best baseline by 9.3%, and on *MeetingBank*, it achieves over 7.7% improvement. In particular, except for PMoE, which employs replay, God-MoE demonstrates stronger resistance to forgetting than all other baselines. Nevertheless, our performance on the *NumGLUE-cm* is slightly lower than the best result, mainly due to the limited number of test samples (only 81), which leads to unstable evaluation. Additionally, we evaluate the model's general performance on five well-known LLM benchmarks with the Open-Compass toolkit (Contributors 2023) after continual learning, as shown in Table 2. The results show that GoD-MoE maintains competitive performance with most baselines. Detailed results are provided in Appendix.

We also compare with several direct tuning methods for LLMs, including full-parameter fine-tuning (Full-FT), standard LoRA (LoRA), LoRA with 1% data replay (LoRA-RE), and multi-expert LoRA tuning (MoE-LoRA). The results show that Full-FT achieves better performance than standard LoRA tuning, but it requires more computation. LoRA-RE, which incorporates replay data, effectively mitigates forgetting but raises concerns about data privacy and storage in real-world applications. Simply adding more experts, as MoE-LoRA, brings limited improvement and still suffers from catastrophic forgetting, suggesting that static expert allocation is insufficient for continual learning. In contrast, our proposed GoD-MoE achieves a favorable balance between plasticity, stability, and privacy by dynamically expanding experts without relying on explicit replay.

Further Analysis

Knowledge Retaining and Acquisition Evaluating a model's ability to retain and acquire knowledge is important for continual instruction tuning. A desirable model should not only preserve previously acquired information but also leverage past knowledge to improve current learning during sequential training. Figure 3 presents the trajectory of task performance across training, which reflects the model's ability to retain knowledge. The results indicate that GoD-MoE experiences the least performance degradation from the current task to the final evaluation. This suggests that freezing experts effectively preserve crucial knowledge and mitigates catastrophic forgetting. In contrast, LoRA and MoE-LoRA suffer more forgetting due to knowledge overwrite, resulting in degraded performance on previous tasks.

Figure 4 reports the performance of knowledge acquisition, which shows how well the model performs on the current task. GoD-MoE outperforms the *single-task* setting on

		C-STANCE	FOMC	MeetingBank	Py150	ScienceQA	NumGLUE-cm	NumGLUE-ds	20Minuten	Avg	BWT
GoD-MoE	Last	49.1	68.8	48.6	54.8	88.1	38.3	60.3	37.6	55.7	-3.6
	Params(%)	0.26	0.21	0.23	0.22	0.16	0.14	0.12	0.22		
GoD-MoE(AAE)	Last	48.6	69.8	49.2	55.2	87.9	40.7	60.9	36.6	56.1	-3.5
	Params(%)	0.26	0.21	0.23	0.22	0.23	0.27	0.31	0.34		

Table 3: Comparison of GoD-MoE and a naive always-add-expert (AAE) variant on performance and training parameters.

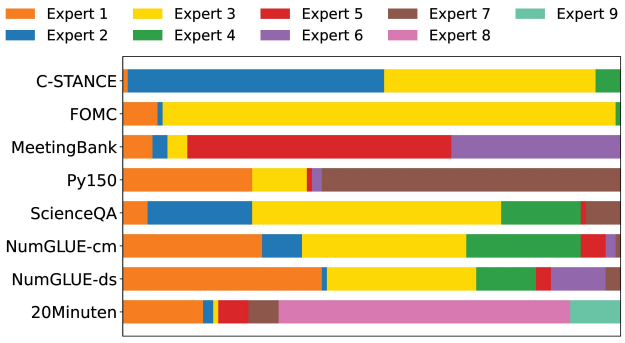

Figure 6: Visualization of expert utilization for various tasks

Metrics	Order-1	Order-2	Order-3
Avg	55.7	56.4	55.0
BWT	-3.6	-3.7	-3.7

Table 4: Evaluation results of different task orders.

most tasks, indicating that previously acquired knowledge can be effectively transferred to benefit new task learning. Moreover, compared to LoRA and MoE-LoRA, the proposed expert expansion strategy mitigates knowledge overwrite and consistently improves current task performance.

Analysis of Expert Reuse and Expansion Figure 5 presents the similarity score (computed by Eq. (6) and (7)) and the number of experts for each task throughout training. The results suggest that new experts are mostly added during the early stages. As training progresses, the existing experts can be reused in various combinations to handle new tasks, reducing the need for further expansion. Specifically, after the first four tasks, the number of experts increases to seven. These experts generalize well to subsequent tasks such as *ScienceQA*, *NumGLUE-cm*, and *NumGLUE-ds*, with no need for new experts. However, for the *20Minuten* task, which is in Germany, the existing experts fail to provide sufficient coverage, leading to the addition of two new experts.

Visualization To analyze expert utilization, Figure 6 visualizes the selection frequency of each expert for each task. The length of the colored bars indicates how frequently each expert is selected. The results demonstrate distinct expert preferences for different tasks, suggesting that different tasks benefit from distinct sets of experts. Moreover, we observe that similar tasks tend to activate similar experts. For example, *NumGLUE-cm* and *NumGLUE-ds* exhibit nearly identical expert selection patterns, as they are both numerical reasoning tasks. In contrast, tasks with greater differences, such as *ScienceQA* and *20Minuten*, rely on entirely different sets of experts. These results indicate GoD-MoE's ability to reuse relevant experts for related tasks while adapting new experts for diverse ones.

Analysis of Training Efficiency We compare our proposed sparse and adaptive expert expansion strategy with a naive variant that always adds a new expert after each task, denoted as GoD-MoE (AAE). Table 3 reports the performance and the proportion of trainable parameters for each approach under the same task order. The result indicates that our sparse and adaptive strategy achieves comparable performance to the always-add-expert variant, with only a 0.4% degradation, while reducing trainable parameters by 35.3% in the final model. This highlights the effectiveness of our strategy in balancing model capacity and computational efficiency. It also suggests that unnecessary expert expansion can be avoided without sacrificing performance, enabling more scalable and efficient continual learning.

Analysis of Different Task Orders To evaluate the robustness of our method, we perform additional experiments with different task orders (Order-2 and Order-3). As shown in Table 4, GoD-MoE maintains consistent and stable performance regardless of the task order. This indicates that our method is not sensitive to a particular task sequence and can reliably handle variations in continual instruction tuning scenarios. Detailed results are provided in Appendix.

Conclusion

In this work, we propose GoD-MoE, a parameter-efficient framework for continual instruction tuning that adaptively expands and reuses experts with a lightweight Expert Demand Detector. This strategy promotes knowledge sharing across tasks, reduces redundant parameter growth, and mitigates catastrophic forgetting without relying on replay data. Extensive experiments on the TRACE benchmark demonstrate that GoD-MoE achieves state-of-the-art performance, maintaining a favorable balance of plasticity, stability, and privacy. Our findings highlight the potential of expert structure sharing to enhance continual learning for LLMs.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (No. 62272250), the Natural Science Foundation of Tianjin, China (No. 22JCJQJC00150).

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2): 235–256.

Bisk, Y.; Zellers, R.; Gao, J.; Choi, Y.; et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 7432–7439.

Clark, C.; Lee, K.; Chang, M.-W.; Kwiatkowski, T.; Collins, M.; and Toutanova, K. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Contributors, O. 2023. Opencompass: A universal evaluation platform for foundation models.

Ding, N.; Qin, Y.; Yang, G.; Wei, F.; Yang, Z.; Su, Y.; Hu, S.; Chen, Y.; Chan, C.-M.; Chen, W.; et al. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3): 220–235.

Dou, S.; Zhou, E.; Liu, Y.; Gao, S.; Zhao, J.; Shen, W.; Zhou, Y.; Xi, Z.; Wang, X.; Fan, X.; et al. 2023. Loramoe: Revolutionizing mixture of experts for maintaining world knowledge in language model alignment. *arXiv preprint arXiv:2312.09979*, 4(7).

Gal, Y.; and Ghahramani, Z. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, 1050–1059. PMLR.

Gao, C.; Chen, K.; Rao, J.; Sun, B.; Liu, R.; Peng, D.; Zhang, Y.; Guo, X.; Yang, J.; and Subrahmanian, V. 2024. Higher layers need more lora experts. *arXiv preprint arXiv:2402.08562*.

Ge, C.; Wang, X.; Zhang, Z.; Chen, H.; Fan, J.; Huang, L.; Xue, H.; and Zhu, W. 2025. Dynamic Mixture of Curriculum LoRA Experts for Continual Multimodal Instruction Tuning. *arXiv preprint arXiv:2506.11672*.

Guo, H.; Zeng, F.; Xiang, Z.; Zhu, F.; Wang, D.-H.; Zhang, X.-Y.; and Liu, C.-L. 2025. Hide-llava: Hierarchical decoupling for continual instruction tuning of multimodal large language model. *arXiv preprint arXiv:2503.12941*.

Guo, Y.; Xu, S.; Fu, J.; Liu, J.; Dong, C.; and Wang, B. 2024. Q-tuning: Queue-based prompt tuning for lifelong few-shot language learning. *arXiv preprint arXiv:2404.14607*.

Han, X.; Dai, Y.; Gao, T.; Lin, Y.; Liu, Z.; Li, P.; Sun, M.; and Zhou, J. 2020. Continual relation learning via episodic memory activation and reconsolidation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, 6429–6440.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2): 3.

Jacobs, R. A.; Jordan, M. I.; Nowlan, S. J.; and Hinton, G. E. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1): 79–87.

Jin, X.; Lin, B. Y.; Rostami, M.; and Ren, X. 2021. Learn Continually, Generalize Rapidly: Lifelong Knowledge Accumulation for Few-shot Learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, 714–729.

Jung, M. J.; and Kim, J. 2024. PMoE: Progressive Mixture of Experts with Asymmetric Transformer for Continual Learning. *arXiv preprint arXiv:2407.21571*.

Ke, Z.; Liu, B.; Xiong, W.; Celikyilmaz, A.; and Li, H. 2023. Sub-network Discovery and Soft-masking for Continual Learning of Mixed Tasks. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 15090–15107.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13): 3521–3526.

Kornblith, S.; Norouzi, M.; Lee, H.; and Hinton, G. 2019. Similarity of neural network representations revisited. In *International conference on machine learning*, 3519–3529. PMLR.

Lee, S.; Ha, J.; Zhang, D.; and Kim, G. 2020. A neural dirichlet process mixture model for task-free continual learning. *arXiv preprint arXiv:2001.00689*.

Liu, Q.; Wu, X.; Zhao, X.; Zhu, Y.; Xu, D.; Tian, F.; and Zheng, Y. 2023. Moelora: An moe-based parameter efficient fine-tuning method for multi-task medical applications. *CoRR*.

McCloskey, M.; and Cohen, N. J. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, 109–165. Elsevier.

Mermillod, M.; Bugaiska, A.; and Bonin, P. 2013. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects.

Mu, S.; and Lin, S. 2025. A comprehensive survey of mixture-of-experts: Algorithms, theory, and applications. *arXiv preprint arXiv:2503.07137*.

Qin, C.; Chen, C.; and Joty, S. 2023. Lifelong Sequence Generation with Dynamic Module Expansion and Adaptation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 6701–6714.

Qing, P.; Gao, C.; Zhou, Y.; Diao, X.; Yang, Y.; and Vosoughi, S. 2024. AlphaLoRA: Assigning LoRA Experts Based on Layer Training Quality. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 20511–20523.

Razdaibiedina, A.; Mao, Y.; Hou, R.; Khabsa, M.; Lewis, M.; and Almahairi, A. 2023. Progressive prompts: Continual learning for language models. *arXiv preprint arXiv:2301.12314*.

Ren, W.; Li, X.; Wang, L.; Zhao, T.; and Qin, W. 2024. Analyzing and reducing catastrophic forgetting in parameter efficient tuning. *arXiv preprint arXiv:2402.18865*.

Rolnick, D.; Ahuja, A.; Schwarz, J.; Lillicrap, T.; and Wayne, G. 2019. Experience replay for continual learning. *Advances in neural information processing systems*, 32.

Shi, H.; Xu, Z.; Wang, H.; Qin, W.; Wang, W.; Wang, Y.; Wang, Z.; Ebrahimi, S.; and Wang, H. 2024. Continual learning of large language models: A comprehensive survey. *ACM Computing Surveys*.

Suzgun, M.; Scales, N.; Schärli, N.; Gehrmann, S.; Tay, Y.; Chung, H. W.; Chowdhery, A.; Le, Q. V.; Chi, E. H.; Zhou, D.; et al. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Team, G.; Georgiev, P.; Lei, V. I.; Burnell, R.; Bai, L.; Gulati, A.; et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context, 2024. *URL https://arxiv. org/abs/2403.05530*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wang, L.; Zhang, X.; Su, H.; and Zhu, J. 2024. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wang, X.; Chen, T.; Ge, Q.; Xia, H.; Bao, R.; Zheng, R.; Zhang, Q.; Gui, T.; and Huang, X. 2023a. Orthogonal subspace learning for language model continual learning. *arXiv preprint arXiv:2310.14152*.

Wang, X.; Zhang, Y.; Chen, T.; Gao, S.; Jin, S.; Yang, X.; Xi, Z.; Zheng, R.; Zou, Y.; Gui, T.; et al. 2023b. Trace: A comprehensive benchmark for continual learning in large language models. *arXiv preprint arXiv:2310.06762*.

Wang, Y.; Mishra, S.; Alipoormolabashi, P.; Kordi, Y.; Mirzaei, A.; Arunkumar, A.; Ashok, A.; Dhanasekaran, A. S.; Naik, A.; Stap, D.; et al. 2022a. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv preprint arXiv:2204.07705*.

Wang, Z.; Zhang, Z.; Lee, C.-Y.; Zhang, H.; Sun, R.; Ren, X.; Su, G.; Perot, V.; Dy, J.; and Pfister, T. 2022b. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 139–149.

Zheng, J.; Qiu, S.; Shi, C.; and Ma, Q. 2025. Towards lifelong learning of large language models: A survey. *ACM Computing Surveys*, 57(8): 1–35.