Unifying Detection and Adaptation in Task-Free Continual Learning

Dezheng Han[1], Anbang Zhang[2], Zhihao Zhu[1], Shuaishuai Guo[1,*]
[1]Shandong University
[2]The Hong Kong University of Science and Technology
{dezhenghan, 202415670}@mail.sdu.edu.cn,
azhangay@connect.ust.hk, shuaishuai_guo@sdu.edu.cn

Abstract

To mitigate catastrophic forgetting in downstream continual learning (CL) for large language models (LLMs), existing methods typically constrain parameter updates or introduce task-specific adaptation modules. However, these methods often rely on explicit task boundaries during training, limiting their applicability to realistic task-free scenarios. In this paper, we propose a **Fi**sher-guided **uni**fied (**FiUni**) framework for batch-level task detection and parameter-efficient continual adaptation. FiUni is motivated by a key observation about the Fisher information matrix (FIM) of pre-trained models: the orthogonality among the principal subspaces of its Kronecker-Factored Approximate Curvature (K-FAC) approximation, estimated from a small number of downstream task samples, can reflect the similarity between different tasks. Based on this observation, FiUni constructs FIM-derived frozen subspaces to guide low-rank adaptation (LoRA), while matching the Fisher principal subspace of each incoming batch window with historical subspaces. This enables FiUni to adaptively determine whether to reuse existing knowledge, expand a related subspace, or create a new subspace, dynamically balancing knowledge sharing and task isolation. Experiments show that FiUni can effectively infer latent batch-level task affiliations and achieve competitive performance against advanced task-aware CL methods with fewer trainable parameters.

1 Introduction

With the remarkable generalization capabilities of large language models (LLMs) (Raffel et al., 2020; Grattafiori et al., 2024; Yang et al., 2025) across diverse downstream tasks, enabling LLMs to continuously adapt to dynamic environments has emerged as a critical research problem (Shi et al., 2025). However, direct full-parameter fine-tuning for continual learning (CL) (Zhou et al., 2024; Wang et al., 2024) suffers from prohibitive computational and storage costs, and inevitably leads to severe catastrophic forgetting (McCloskey and Cohen, 1989), where the model loses performance on old tasks after learning new ones. Parameter-efficient fine-tuning (PEFT) (Han et al., 2024; Wang et al., 2025), which updates only a small number of original or additional parameters, provides a more feasible solution to this problem. Following this idea, existing parameter-efficient continual learning methods typically manage the low-dimensional parameter update spaces of different tasks, thereby reducing inter-task interference while maintaining fine-tuning efficiency (Wang et al., 2023a; Liao et al., 2026; Biswas et al., 2026).

Despite promising performance in multi-task continual learning, existing methods typically rely on task-aware assumptions, where task boundaries or IDs are available for allocating and switching parameter subspaces. In contrast, real-world continual learning is often task-free, with data arriving in an online stream without explicit boundary annotations or task identities (Aljundi et al., 2019). Thus, the model must autonomously infer latent task structure while adapting its parameters to incoming data. Although recent methods exploit model-internal task-related signals to relax this assumption (Du et al., 2024), they mainly focus on task-label-free forgetting mitigation, rather than explicitly unifying task identification with the isolation and sharing of PEFT subspaces.

To address these issues, we need a meaningful geometric signal that can characterize the relationships among different incoming data and further guide parameter-efficient adaptation. Fisher geometry (ichi Amari, 1998) provides a promising perspective for this purpose. Recent studies have shown that the Kronecker-Factored Approximate Curvature (K-FAC) (Martens and Grosse, 2015)

Code is available at: https://github.com/SSG2019/FiUni. *Corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 15153–15177
October 24-29, 2026 ©2026 Association for Computational Linguistics

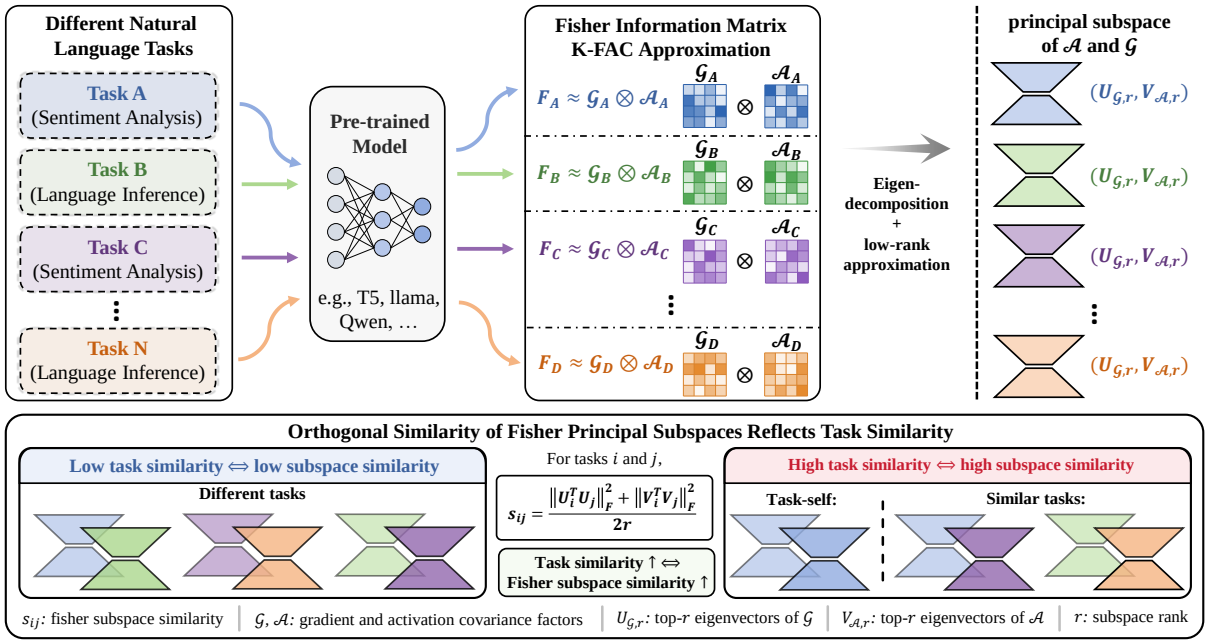

Figure 1: Fisher principal subspaces encode task similarity. For each downstream task, we estimate the K-FAC approximation and extract the top-r principal subspaces of its gradient and activation covariance factors. Higher subspace similarity indicates more similar task geometry, while lower similarity indicates stronger orthogonality.

of the Fisher information matrix (FIM), estimated from a small number of downstream samples of different tasks on a pre-trained model, can serve as effective fixed low-rank update bases to guide LoRA along parameter-sensitive directions (Han and Guo, 2026). This motivates us to further explore whether Fisher principal subspaces can be used to model latent task relationships among different downstream data.

Under this perspective, we discover that in the K-FAC (Martens and Grosse, 2015) approximation of the FIM estimated on a few downstream samples, the orthogonality of the principal subspaces, spanned by the top-r eigenvectors of the gradient and activation covariance factors, naturally reflects the geometric similarity between distinct tasks, as shown in Figure 1. Specifically, data from identical or related tasks tend to exhibit higher Fisher principal subspace similarity, whereas those from unrelated tasks tend towards orthogonality. This indicates that for pre-trained models, the Fisher principal subspace not only characterizes the sensitive update directions for the current data but also serves as a geometric signal to identify the latent task affiliation of each batch in an online data stream. Here, a latent task does not necessarily align perfectly with human-annotated discrete tasks, but rather represents an implicit learning phase perceived by the pre-trained model, characterized by similar learning requirements and adaptation behaviors.

Motivated by the above observation, we propose a Fisher geometry-based unified task-free framework, named FiUni, for the continual parameter-efficient fine-tuning scenario. The key idea of FiUni is to utilize the same Fisher principal subspace signal to simultaneously perform batch-level latent task detection and LoRA subspace construction. Specifically, FiUni maintains a storehouse of historical Fisher subspaces during online training and matches the Fisher principal subspace of each incoming batch against this storehouse. The resulting subspace similarity serves as a unified geometric signal to adaptively determine whether the current batch should reuse an existing subspace, expand a related subspace, or be assigned to a new subspace. Thus, FiUni jointly models latent task detection and parameter-efficient adaptation within a single framework, enabling the model to dynamically balance knowledge sharing and task-specific isolation without relying on explicit task boundaries or IDs.

The main contributions of this work are summarized as follows:

- We reveal the intrinsic task geometry within Fisher/K-FAC principal subspaces. By utilizing only a small number of downstream samples to estimate the Fisher/K-FAC principal subspace, we depict meaningful task-geometric structures, thereby providing an ef-

fective signal for measuring task similarity.

- We propose FiUni, a task-free continual parameter-efficient fine-tuning framework that unifies latent task shift detection and LoRA subspace construction via a single geometric signal. Furthermore, we design an adaptive decision mechanism to dynamically balance knowledge sharing and task isolation based on the matching degree between current data and historical Fisher subspaces.

- Experimental results demonstrate that FiUni can effectively infer the latent task affiliation of each incoming batch without relying on task boundaries or IDs, and achieve competitive performance compared to state-of-the-art task-aware continual learning baselines while requiring fewer trainable parameters.

2 Related Work

CL (Zhou et al., 2024; Wang et al., 2024) aims to enable models to sequentially obtain knowledge from continuous data streams or tasks while preserving performance on previously learned information. The primary challenge in this paradigm is catastrophic forgetting, i.e., adapting to new tasks tends to overwrite or disrupt parameter representations crucial for prior knowledge (French, 1999; McCloskey and Cohen, 1989). To mitigate this issue, existing research broadly categorizes CL approaches into regularization-based, replay-based, and dynamic architecture-based methods. Regularization-based techniques (Kirkpatrick et al., 2017; Zenke et al., 2017; Aljundi et al., 2018) typically preserve crucial parameters of previous tasks by restricting updates on new tasks. Replay-based strategies (Lopez-Paz and Ranzato, 2017; Shin et al., 2017) alleviate forgetting by saving a small number of old task samples or generating historical task samples during the training procedure of new tasks. Moreover, dynamic architecture-based schemes (Lee et al., 2017; Li et al., 2019) progressively expand the network topology to provide additional capacity for new knowledge. Recently, the development of LLMs has shifted the research focus of CL from traditional lightweight classifiers to massive pre-trained models (Shi et al., 2025; Wu et al., 2024).

This shift makes parameter efficiency a central concern. PEFT (Han et al., 2024; Wang et al., 2025) provides a feasible approach by freezing most pre-trained parameters and introducing only a small number of trainable parameters for downstream adaptation. Typical approaches include Adapters (Houlsby et al., 2019), Prefix Tuning (Li and Liang, 2021), and LoRA (Hu et al., 2022). LoRA introduces low-rank update branches into weight matrices, and these updates can be merged back after training without additional inference latency. Building on this low-rank formulation, recent works have further attempted to impose pre-defined update subspaces on LoRA to reduce trainable parameters while maintaining adaptation performance (Kopiczko et al., 2024; Bałazy et al., 2025; Gao et al., 2024). For instance, FiLoRA (Han and Guo, 2026) utilizes the K-FAC (Martens and Grosse, 2015) approximation of the FIM from a small number of downstream samples, to extract principal subspaces as fixed low-rank bases, thereby guiding LoRA to perform efficient adaptation in parameter-sensitive directions. This indicates that the Fisher principal subspace can characterize the important update directions of the current task-specific parameters and provide a more geometrically meaningful subspace selection for efficient parameter fine-tuning.

In CL scenarios, PEFT provides a natural way to manage task-specific adaptation parameters, which can be independently stored, composed, or isolated. Building upon this property, existing continual PEFT methods typically enforce orthogonal constraints to separate the low-rank update spaces of different tasks, thereby reducing inter-task interference (Wang et al., 2023a; Qiao and Mahdavi, 2024). Furthermore, recent methods have moved beyond strict isolation and explicitly model shared and task-specific components across tasks to improve transfer ability (Liao et al., 2026; Biswas et al., 2026). Although current continual PEFT schemes demonstrate strong performance in task-aware settings, their reliance on explicit task boundaries or task IDs for module management limits their applicability to task-free online scenarios. While recent task-label-free methods have explored model-internal signals to reduce the dependence on task identities (Du et al., 2024), they mainly focus on update modulation rather than explicitly coordinating latent task discovery with parameter-efficient subspace organization. Therefore, how to unify task-free latent structure identification and adaptive PEFT subspace management remains underexplored.

The intrinsic difference between FiUni and the above methods lies in its ability to determine the

allocation of incoming knowledge at the batch level within a task-free setting. Furthermore, rather than passively mitigating inter-task interference through auxiliary regularization terms during the training process, FiUni actively confines parameter updates to low-rank subspaces determined by Fisher geometry during the subspace construction stage. Importantly, unlike methods that depend on explicit task boundaries to dictate knowledge sharing and isolation strategies, the mechanisms of knowledge reuse and task isolation in FiUni emerge naturally from the Fisher geometry induced by the pre-trained model on a minimal set of downstream samples, rather than being explicitly specified by human-annotated task partitions.

3 Methodology

3.1 Fisher Subspace Similarity

We first introduce how FiUni measures the geometric similarity between different downstream data batches through Fisher principal subspaces. Consider a linear layer in a pre-trained model with weight matrix $W \in \mathbb{R}^{m\times n}$. For a data window $\mathcal{B}$, the Fisher information matrix of this layer can be approximated by K-FAC (Martens and Grosse, 2015) as the Kronecker product of two covariance factors: $F_W \approx \mathcal{G} \otimes \mathcal{A}$, where

$$\mathcal{A} = \mathbb{E}_{x\sim\mathcal{B}}[xx^\top], \mathcal{G} = \mathbb{E}_{x\sim\mathcal{B}}[\delta\delta^\top]. \tag{1}$$

Here, x denotes the input activation of the layer, and δ denotes the back-propagated gradient with respect to the layer output. The activation covariance factor $\mathcal{A}$ captures the dominant directions in the input feature space, while the gradient covariance factor $\mathcal{G}$ captures the dominant sensitive directions in the output gradient space. Together, they characterize the Fisher geometry induced by the current data on the parameter updates of this layer.

To obtain a compact task representation, we perform eigendecomposition on $\mathcal{A}$ and $\mathcal{G}$, and select the top-r_{det} eigenvectors corresponding to the largest eigenvalues, where r_{det} denotes the detection rank used for Fisher subspace matching:

$$V_{\mathcal{B}} = \text{TopEig}(\mathcal{A}, r_{\text{det}}), U_{\mathcal{B}} = \text{TopEig}(\mathcal{G}, r_{\text{det}}), \tag{2}$$

where $V_{\mathcal{B}} \in \mathbb{R}^{n\times r_{\text{det}}}$ and $U_{\mathcal{B}} \in \mathbb{R}^{m\times r_{\text{det}}}$. We refer to $(U_{\mathcal{B}}, V_{\mathcal{B}})$ as the Fisher principal subspace of the current data window.

For two data windows $\mathcal{B}_i$ and $\mathcal{B}_j$, we measure their geometric similarity by the overlap between their Fisher principal subspaces. Specifically, the gradient-side and activation-side subspace similarities are defined as

$$s_U(\mathcal{B}_i, \mathcal{B}_j) = \frac{1}{r_{\text{det}}} \left\| U_i^\top U_j \right\|_F^2, \tag{3}$$

$$s_V(\mathcal{B}_i, \mathcal{B}_j) = \frac{1}{r_{\text{det}}} \left\| V_i^\top V_j \right\|_F^2. \tag{4}$$

The final Fisher subspace similarity is defined as

$$s(\mathcal{B}_i, \mathcal{B}_j) = \frac{s_U(\mathcal{B}_i, \mathcal{B}_j) + s_V(\mathcal{B}_i, \mathcal{B}_j)}{2}. \tag{5}$$

For a multi-layer model, we compute the above similarity over multiple selected target layers and modules, and average them as the overall task similarity:

$$s_{ij} = \frac{1}{|\mathcal{L}|} \sum_{\ell\in\mathcal{L}} s^{(\ell)}(\mathcal{B}_i, \mathcal{B}_j), \tag{6}$$

where $\mathcal{L}$ denotes the set of layers and modules used for Fisher subspace estimation. This similarity can be interpreted as the average principal-direction overlap between two Fisher geometries. A larger s_{ij} suggests that two data windows may come from the same or related latent task phases, while a smaller value indicates stronger orthogonality and may imply the emergence of a new latent phase. Therefore, this metric can be used to detect latent task changes in an online data stream by measuring task similarity between different data windows.

3.2 FiUni Framework Design

Based on the observation that Fisher principal subspaces reflect task similarity, we propose FiUni, a unified task-free continual learning framework for latent task detection and parameter-efficient adaptation, as shown in Figure 2. FiUni considers an online data stream where training data arrive sequentially in batches:

$$\mathcal{D} = \{\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_n, \ldots\}, \tag{7}$$

During training, we maintain a historical Fisher subspace pool:

$$\mathcal{S} = \{(U_k, V_k, R_k)\}_{k=1}^{K}, \tag{8}$$

where K denotes the number of discovered latent phases. For the k-th phase, U_k and V_k are frozen bases derived from Fisher/K-FAC principal subspaces, and R_k is the corresponding trainable core

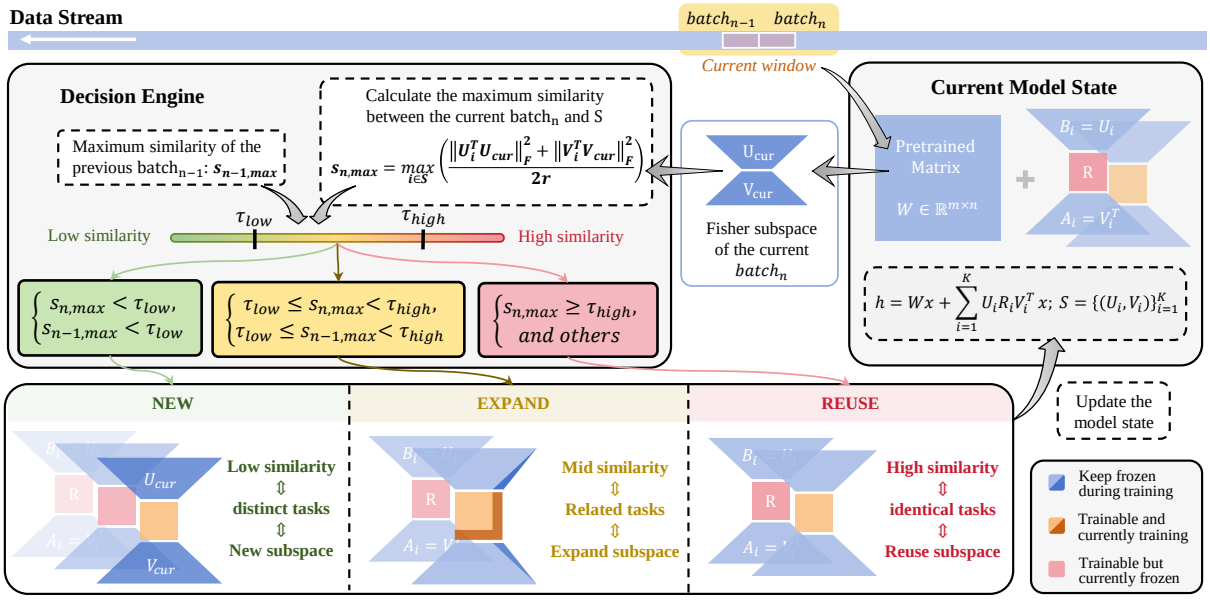

Figure 2: Overview of FiUni. For each incoming batch in a task-free stream, FiUni estimates its Fisher principal subspace and matches it with the historical subspace pool. Based on the subspace similarity, FiUni adaptively decides to create a New subspace, Expand a related subspace, or Reuse an existing one.

matrix. For a given layer, FiUni represents the effective model parameter as

$$W_{\text{eff}} = W_0 + \sum_{k=1}^{K} U_k R_k V_k^\top, \tag{9}$$

where W_0 is the frozen pre-trained weight. Unlike standard LoRA, which directly learns a full low-rank decomposition, FiUni fixes the left and right update bases using Fisher principal subspaces and only trains the compact intermediate matrix R_k, thereby restricting parameter updates to Fisher-sensitive directions of the current data.

When a new data window $\mathcal{B}_n$ arrives, FiUni first estimates its Fisher principal subspace by extracting the top-r principal directions $(U_{\text{cur}}, V_{\text{cur}})$ for subsequent LoRA subspace construction.

It then computes the similarity between the current subspace and all historical subspaces using the top-r_{det} principal directions, with $r_{\text{det}} \le r$, and takes the maximum value as the matching degree between the current window and historical knowledge:

$$s_{n,\max} = \max_{k \in \{1,\dots,K\}} s\left((U_{\text{cur}}, V_{\text{cur}}), (U_k, V_k)\right). \tag{10}$$

This maximum similarity reflects whether the current data can be explained by existing Fisher subspaces. A larger $s_{n,\max}$ indicates that the current data is highly related to a historical phase, while a smaller value suggests that the current data may correspond to a new latent task phase.

Based on the matching degree between the current window and historical subspaces, FiUni adopts a three-state decision mechanism: REUSE, EXPAND, NEW. This mechanism is controlled by two thresholds, τ_{low} and τ_{high}.

When $s_{n,\max} \ge \tau_{\text{high}}$, FiUni regards the current data as highly matched with an existing phase, and thus reuses the corresponding LoRA subspace by continuing to update its core matrix R_k. When $\tau_{\text{low}} \le s_{n,\max} < \tau_{\text{high}}$, FiUni treats the current data as related to an existing phase with moderate drift. It therefore expands the most similar FiLoRA subspace by removing the overlapping components of $(U_{\text{cur}}, V_{\text{cur}})$, orthogonalizing the remaining directions, and supplementing them with random directions to increase the subspace rank by Δr. When $s_{n,\max} < \tau_{\text{low}}$, FiUni considers the current data geometrically distinct from historical phases and allocates a new LoRA subspace using the current Fisher principal subspace:

$$U_{K+1} = U_{\text{cur}}, V_{K+1} = V_{\text{cur}}, \tag{11}$$

and initializes a new trainable core matrix R_{K+1}. The new tuple $(U_{K+1}, V_{K+1}, R_{K+1})$ is then added to the historical subspace pool.

To improve online detection stability and reduce false triggers caused by sample noise, FiUni adopts a two-window confirmation mechanism. For two

consecutive batches/windows, FiUni performs the corresponding REUSE or NEW operation only when both satisfy the same decision condition, making online task-shift detection more robust.

3.3 Geometric Reuse and Isolation in FiUni

Unlike methods that rely on given task boundaries to actively isolate adaptation modules and predefine shared or isolated parameters, knowledge sharing and task isolation in FiUni both arise from Fisher geometry itself, and parameter updates are restricted to the corresponding geometric subspaces.

Taking the output-gradient Fisher subspace U as an example, for the current historical subspace set $\mathcal{S}$, the common overlapping component among these subspaces corresponds to reusable shared directions:

$$U_{\mathcal{S}}^{\text{share}} = \bigcap_{k=1}^{K} U_k. \tag{12}$$

For any subspace U_i and the remaining subspace pool $U_{\neg i}$, the orthogonal component corresponds to more task-specific isolated directions for the latent task:

$$U_i^{\text{iso}} = (I - U_{\neg i} U_{\neg i}^{\top}) U_i. \tag{13}$$

The overlapping components support knowledge reuse across related phases, while the orthogonal residuals provide isolation for new or drifting knowledge. Therefore, FiUni does not simply switch modules after detecting task boundaries. Instead, it performs geometric matching at the batch level and adaptively balances reuse, expansion, and isolation throughout the online data stream.

In practice, estimating Fisher statistics and performing eigendecomposition for all parameter blocks would introduce considerable computational overhead, while our ablation experiments show that the discriminative ability of Fisher subspace similarity is not very sensitive to the specific choice of layers or modules, but is mainly affected by the number of samples used for estimation and the r_{det}. Therefore, FiUni performs subspace matching only on a small number of selected layers. This enables FiUni to perform efficient online detection with limited overhead, making it practical for continual parameter-efficient fine-tuning of large language models.

4 Experiment

4.1 Benchmarks and Data Setup

We evaluate FiUni on three CL benchmarks. Unlike conventional task-aware continual learning, we focus on the more challenging task-free online setting, where training data arrive sequentially as a batch stream and the model has no access to explicit task boundaries or task IDs during training.

Standard CL Benchmark (SC) consists of four text classification datasets: AG News, Amazon Reviews, DBpedia, and Yahoo Answers (Zhang et al., 2015). Following prior work (Wang et al., 2023a), we organize them into three task orders, denoted as Orders 1–3, to evaluate continual adaptation over short classification sequences.

Long Sequence Benchmark (LS) extends SC to a longer and more diverse stream with 15 tasks (Razdaibiedina et al., 2023). We sample a fixed number of training examples for each task and construct three task orders, denoted as Orders 4–6. Compared with SC, LS contains more heterogeneous task types and more complex task transitions.

TRACE Benchmark (Wang et al., 2023b) contains diverse LLM tasks, including multiple-choice QA, multilingual understanding, code generation, and mathematical reasoning, denoted as Order 7.

4.2 Task Similarity

We first examine the core observation of this work: Fisher subspace similarity s, estimated from a small number of downstream samples, can reflect geometric similarity between tasks. Specifically, we use T5-base on the LS benchmark and sample 32 examples from each task to compute the pairwise Fisher subspace similarity s defined in Section 3.1. We then analyze whether the resulting similarity structure is consistent with task semantics or task families.

As shown in Figure 3 (a), the Fisher subspace similarity matrix exhibits clear task structure. Data from identical or related tasks usually show higher subspace similarity, while unrelated tasks tend to be more orthogonal. For example, MNLI, CB, and RTE, which are all related to language inference, show relatively high similarity. Among sentiment-related tasks, IMDB, SST-2, Yelp, and Amazon are all sentiment-oriented, but Yelp and Amazon exhibit stronger similarity, likely because both are rating-style review tasks.

We further evaluate the within-task consistency of this signal through repeated sampling, as shown

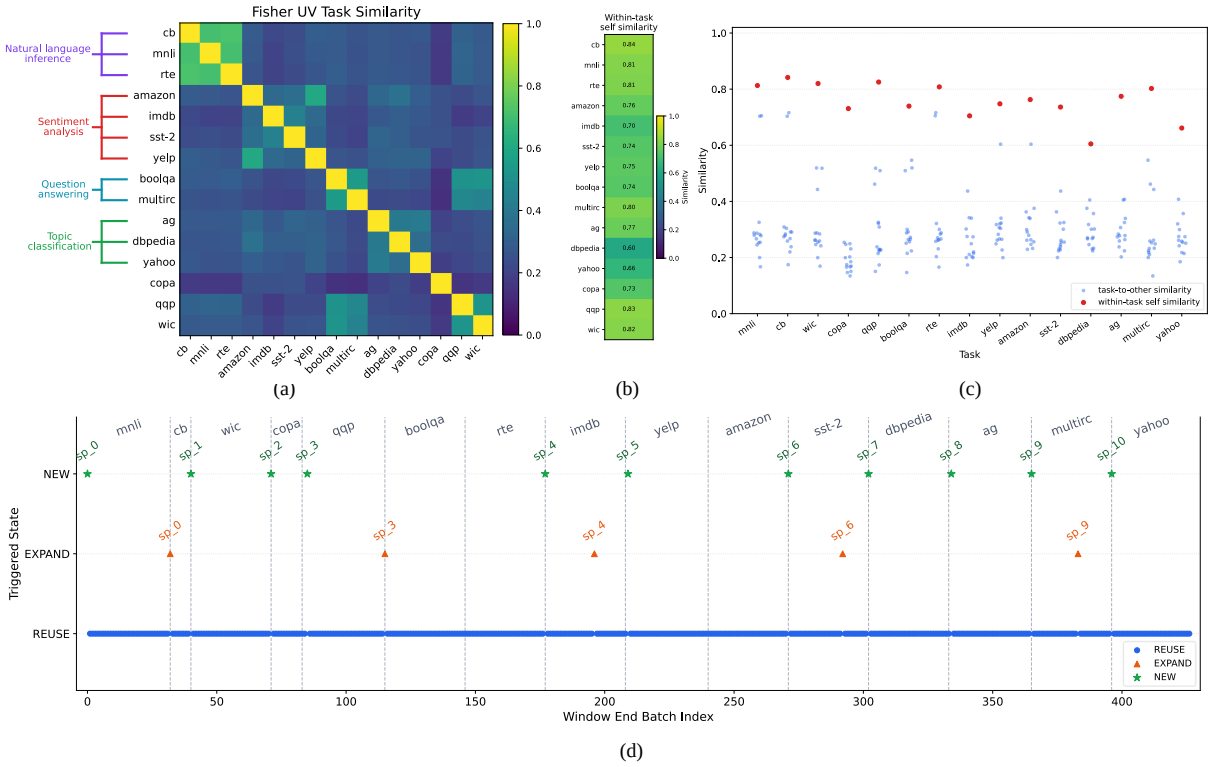

Figure 3: Empirical analysis of Fisher subspace similarity and FiUni decisions on T5-base. Panels (a)–(c) estimate Fisher subspaces using 32 samples per task over all linear layers with $r_{\text{det}} = 8$: (a) shows cross-task Fisher subspace similarity with task-family annotations, (b) shows within-task self-similarity from two independent samplings using a compact $N \times 1$ heatmap, and (c) compares within-task similarity with task-to-other similarity. Panel (d) shows the batch-level decisions produced by FiUni on a task-free stream using 32-shot Fisher estimation, selected Q/K/V/O layers, and thresholds $\tau_{\text{low}} = 0.65$ and $\tau_{\text{high}} = 0.85$.

in Figure 3 (b). For each task, we independently sample two groups of examples and estimate their Fisher principal subspaces separately. The results show that two subsets from the same task usually produce highly similar Fisher subspaces, indicating that samples within the same task tend to share consistent Fisher-geometric characteristics. Finally, Figure 3 (c) compares within-task self-similarity and task-to-other similarity. Most tasks exhibit substantially higher self-similarity than cross-task similarity. In particular, similarities between identical or related tasks are mostly concentrated in the range of 0.6–0.9, whereas low-similarity or unrelated task pairs mostly fall below 0.6. This suggests that appropriate thresholds can separate similar tasks from unrelated ones, supporting the feasibility of Fisher subspace similarity as a batch-level signal for latent task identification.

4.3 Task Detection

After verifying that Fisher subspaces reflect task similarity, we further analyze the batch-level decision process of FiUni in an online task-free stream. We take up to 1,000 samples from each task in the LS benchmark and concatenate them into a sequential training stream. During detection, the model has no access to task IDs or task boundaries, and makes decisions batch by batch.

Figure 3 (d) shows the batch-level decision trajectory of FiUni in the task-free stream. When the stream remains within the same task or moves to a related task, FiUni usually tends to reuse an existing subspace or perform a lightweight expansion. When the stream enters a more distinct stage, FiUni can trigger the allocation of a new subspace. For example, within the SST-2 stage, FiUni expands the current subspace when the internal data variation becomes larger. In contrast, when the stream transitions from CB to WiC or from WiC to COPA, FiUni creates a new subspace.

To complement the qualitative decision trajectory in Figure 3(d), we further quantify the batch-level detection behavior of FiUni. We emphasize that this evaluation is only used for post-hoc analysis: during online training and detection, FiUni does not access task IDs or task boundaries. Man-

	Methods	Standard CL Benchmark (SC)				Long Sequence Benchmark (LS)				TRACE
		Order 1	Order 2	Order 3	Avg	Order 4	Order 5	Order 6	Avg	Order 7
T5-Large	SeqLoRA†	39.5	31.9	46.6	39.3	4.9	3.5	4.2	4.2	12.1
	+MIGU† (Du et al., 2024)	74.8	71.6	73.5	73.3	66.9	64.8	51.8	61.2	-
	SeqLoRAReplay	74.0	73.1	73.0	73.3	74.2	72.7	73.9	73.6	34.0
	IncLoRA	63.4	62.2	65.1	63.6	63.0	57.9	60.4	60.5	-
	EWC (Kirkpatrick et al., 2017)	46.3	45.3	52.1	47.9	44.9	44.0	45.4	44.8	-
	L2P (Wang et al., 2022)	60.3	61.7	61.1	60.7	57.5	53.8	56.9	56.1	-
	LFPT5 (Qin and Joty, 2022)	67.6	72.6	77.9	72.7	70.4	68.2	69.1	69.2	-
	O-LoRA (Wang et al., 2023a)	73.2	72.4	70.4	72.0	69.9	68.5	65.3	67.9	23.1
	+ MIGU (Du et al., 2024)	73.5	71.4	70.0	71.6	65.4	65.2	65.2	65.3	-
	SpaRTA (Liao et al., 2026)	73.7	70.5	**73.8**	72.7	**71.5**	**70.5**	68.0	**70.0**	16.7
	+ Replay	77.0	75.6	75.2	75.9	75.6	73.2	74.1	74.3	36.5
	FiUni (ours)†	**75.6**	**76.5**	73.0	**75.0**	67.1	67.0	**70.84**	68.3	**32.9**
LLaMA-3.1-8B	SeqLoRA†	75.88	74.40	74.35	74.86	67.81	65.93	63.80	65.85	28.23
	O-LoRA (Wang et al., 2023a)	69.39	67.58	71.44	69.46	69.54	64.42	66.50	66.82	28.45
	SpaRTA (Liao et al., 2026)	76.10	75.69	75.52	75.77	71.34	70.77	72.95	71.69	31.33
	+ Replay	77.56	77.01	76.03	76.83	73.10	73.05	74.17	73.44	34.16
	SeqLoRAReplay	77.02	76.53	76.19	76.58	72.03	72.96	75.38	73.46	33.02
	ELLA (Biswas et al., 2026)	77.80	77.20	**77.70**	77.57	72.87	72.84	76.82	74.18	33.29
	FiUni (ours)†	**78.00**	**78.20**	77.49	**77.91**	**75.74**	**74.74**	**77.50**	**75.99**	**55.21**

Table 1: Comparison results on Standard CL Benchmark (SC), Long Sequence Benchmark (LS), and TRACE. FiUni results are averaged over three random seeds. Methods marked with † do not rely on task boundaries during training. Gray-colored methods use replay data, generated pseudo-data, or external memory during continual learning. **Bold** indicates the best result, and underline indicates the second-best result among non-replay methods.

Order	Precision ↑	Recall ↑	F1 ↑	Delay ↓	False Trigger ↓	Stable REUSE ↑
Order 4	1.000	0.786	0.880	1.364	0.000	1.000
Order 5	1.000	0.786	0.880	1.545	0.000	1.000
Order 6	1.000	0.786	0.880	1.455	0.000	1.000
Avg.	1.000	0.786	0.880	1.455	0.000	1.000

Table 2: Quantitative evaluation of FiUni's batch-level detection behavior on LLaMA-3.1-8B under the LS benchmark.

ual dataset boundaries are used only after the online decision process has been completed.

We adopt a strict dataset-level boundary protocol. Each transition between two adjacent datasets in the stream is treated as a reference boundary. If FiUni triggers EXPAND or NEW within a fixed detection window after a reference boundary, we regard it as a successful structural response to this transition. We report Boundary Precision, Boundary Recall, F1, Detection Delay, False Trigger Rate, and Stable-region REUSE Rate. Boundary Precision measures whether FiUni's structural responses correspond to reference boundaries, Boundary Recall measures whether reference boundaries are covered by FiUni, and Detection Delay measures the number of windows from a boundary to the first structural response. False Trigger Rate and Stable-region REUSE Rate evaluate whether unnecessary EXPAND, NEW, or active-subspace switching occurs in stable regions away from boundaries.

As shown in Table 2, FiUni achieves a Boundary Precision of 1.000 and a False Trigger Rate of 0.000 across all three orders, indicating that its structural responses are well aligned with reference transitions and do not introduce extra false triggers. Meanwhile, the average Boundary Recall is 0.786 and the average Detection Delay is 1.455 windows, showing that FiUni can cover most dataset-level transitions with a short response delay. The Stable-region REUSE Rate remains 1.000, which further indicates that FiUni maintains stable reuse behavior away from task transitions.

We further note that the strict dataset-level protocol is conservative for FiUni, because FiUni models latent task phases rather than manual dataset identities. For related task families, continuing to reuse an existing subspace can be consistent with FiUni's latent-task modeling objective. Therefore, we additionally report a family-adjusted latent-consistency evaluation, where reuse within a related task family

Order	Precision ↑	Recall ↑	F1 ↑
Order 4	1.000	1.000	1.000
Order 5	1.000	1.000	1.000
Order 6	1.000	1.000	1.000
Avg.	1.000	1.000	1.000

Table 3: Family-adjusted latent-consistency evaluation on the LS benchmark.

is treated as a consistent latent decision.

Under this family-adjusted protocol, FiUni reaches 1.000 Precision, Recall, and F1 across all three orders. This shows that the unmatched boundaries under the strict protocol mainly correspond to latent-consistent knowledge reuse within related task families, rather than random detection failures.

We also observe that FiUni's decisions do not always coincide with human-defined task boundaries. For some related tasks, FiUni may reuse or expand an existing subspace instead of immediately creating a new one. For instance, when the stream moves from MNLI to CB, FiUni expands the subspace associated with MNLI rather than allocating a new subspace. During the Amazon stage, FiUni directly reuses or continues training on the subspace previously triggered by Yelp. This is consistent with our definition of latent tasks. Such behavior allows FiUni to avoid mechanically allocating a separate subspace for every dataset-level task, reducing redundant parameter growth while promoting knowledge reuse across related tasks.

4.4 Overall Results

To verify the effectiveness of FiUni in detecting latent tasks, we compare it with multiple task-aware continual learning methods. Since task boundaries are unavailable in our stream setting, we only consider overall accuracy (Chaudhry et al., 2018) $\mathrm{OA}_T = \frac{1}{T}\sum_{t=1}^{T} a_{T,t}$, where $a_{T,t}$ denotes the performance on task t after training on the complete stream. We do not consider FWT (Lopez-Paz and Ranzato, 2017) or BWT (Ke and Liu, 2023), as these metrics rely on task boundaries to measure forgetting and knowledge transfer.

As shown in Table 1, FiUni achieves strong results across different benchmarks and backbone models, and outperforms strict orthogonal isolation methods such as O-LoRA in all settings. This demonstrates the effectiveness of using fixed Fisher subspaces for task sharing and isolation. We observe that FiUni brings relatively limited improvement on LS with T5-Large. A possible reason is that T5-Large has a lower hidden dimension, and to keep the FiLoRA parameter scale competitive with standard LoRA, the available rank capacity can be consumed earlier in the long task stream, thereby restricting the expansion space for later phases.

In contrast, LLaMA-3.1-8B provides a larger representation space and richer subspace capacity, where FiUni achieves more consistent gains across the three benchmarks. Overall, without using task boundaries or task IDs, FiUni still matches or surpasses state-of-the-art task-aware continual PEFT methods. This shows that latent task differences can be effectively captured by the Fisher geometry of downstream data on the pre-trained model, enabling effective continual adaptation even without explicit task information.

5 Conclusion

We propose FiUni, a Fisher geometry-based framework for task-free continual parameter-efficient fine-tuning in online data streams. FiUni unifies batch-level latent task identification and LoRA subspace construction through Fisher principal subspaces, enabling adaptive reuse, expansion, and allocation of LoRA subspaces without explicit task boundaries or task IDs. Experiments demonstrate that Fisher subspace similarity captures meaningful and stable task geometry, and that FiUni achieves competitive performance against strong task-aware baselines with substantially fewer trainable parameters. Our results highlight Fisher geometry as an effective and practical signal for scalable task-free continual adaptation of LLMs.

6 Limitations

FiUni still has several limitations. First, the detection stage estimates Fisher/K-FAC statistics on the pre-trained model, which introduces an extra forward and backward pass. Although this cost is moderate because detection uses only a few samples and selected layers, it may still be noticeable in large-scale online training. A promising future direction is to further study the relationship between task streams and the Fisher factors estimated on the pre-trained model combined with already learned LoRA modules, so that detection can potentially be performed with fewer additional passes or reused statistics. Second, computing the activation and gradient covariance factors also introduces additional memory overhead, since activations and gradients need to be cached for selected modules.

Although FiUni avoids constructing the full Fisher matrix, more memory-efficient estimation strategies, such as streaming covariance updates, low-precision statistics, or randomized eigendecomposition, could further improve scalability. Finally, due to computational and memory constraints, our experiments are limited to models up to 8B parameters. We therefore have not fully assessed the behavior and overhead of FiUni on larger-scale models such as 70B LLMs. Further scaling experiments would be needed to verify its efficiency and robustness in such settings.

7 Ethical Considerations

FiUni is a method for parameter-efficient continual adaptation and does not introduce new data collection, human annotation, or user-facing decision procedures. All experiments are conducted on publicly available benchmarks following standard research protocols. Since FiUni does not require storing previous task samples for replay, it does not add extra data-retention requirements beyond the training data used in each benchmark. As with standard fine-tuning methods for large language models, practical deployment should follow the data usage policies and safety requirements of the target application.

8 AI writing statement

This paper utilized AI assistance for language polishing of the manuscript, including vocabulary correction and spell checking.

Acknowledgements

This work was supported in part by Shandong University Research Project on Laboratory Development and Management under Grant sy20261301, and in part by Shandong Provincial Natural Science Foundation under Grant ZR2025MS983. We thank the anonymous reviewers for their constructive feedback.

References

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. 2018. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision*.

Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. 2019. Task-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11254–11263.

Klaudia Bałazy, Mohammadreza Banaei, Karl Aberer, and Jacek Tabor. 2025. LoRA-XS: Low-rank adaptation with extremely small number of parameters. *Computing Research Repository*, arXiv:2405.17604. Version 3.

Shristi Das Biswas, Yue Zhang, Anwesan Pal, Radhika Bhargava, and Kaushik Roy. 2026. ELLA: Efficient lifelong learning for adapters in large language models. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1907–1924, Rabat, Morocco.

Arslan Chaudhry, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. 2018. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 556–572, Munich, Germany. Springer.

Wenyu Du, Shuang Cheng, Tongxu Luo, Zihan Qiu, Zeyu Huang, Ka Chun Cheung, Reynold Cheng, and Jie Fu. 2024. Unlocking continual learning abilities in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6503–6522, Miami, Florida, USA.

Robert M. French. 1999. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135.

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.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The Llama 3 herd of models. *Computing Research Repository*, arXiv:2407.21783.

Dezheng Han and Shuaishuai Guo. 2026. FiLoRA: Parameter-efficient fine-tuning with fisher information-guided low-rank adaptation. *IEEE Signal Processing Letters*, 33:604–608.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. Parameter-efficient fine-tuning for large models: A comprehensive survey. *Computing Research Repository*, arXiv:2403.14608.

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 *Proceedings of the 36th International Conference on Machine Learning*, pages 2790–2799, Long Beach, USA.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *Proceedings of the 10th International Conference on Learning Representations*, pages 12513–12525, Virtual.

Shun ichi Amari. 1998. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276.

Zixuan Ke and Bing Liu. 2023. Continual learning of natural language processing tasks: A survey. *Computing Research Repository*, arXiv:2211.12701.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Dawid J. Kopiczko, Tijmen Blankevoort, and Yuki M. Asano. 2024. VeRA: Vector-based random matrix adaptation. In *Proceedings of the 12th International Conference on Learning Representations*, pages 37947–37967, Vienna, Austria.

Jeongtae Lee, Jaehong Yoon, Eunho Yang, and Sung Ju Hwang. 2017. Lifelong learning with dynamically expandable networks. *Computing Research Repository*, arXiv:1708.01547.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Virtual.

Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. 2019. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3925–3934.

Huanxuan Liao, Shizhu He, Yupu Hao, Jun Zhao, and Kang Liu. 2026. Spectral disentanglement: Rank-aware task adaptation for rehearsal-free continual learning in LLMs. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, San Diego, California.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, volume 30.

James Martens and Roger Grosse. 2015. Optimizing neural networks with kronecker-factored approximate curvature. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2408–2417, Lille, France.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In Gordon H. Bower, editor, *Psychology of Learning and Motivation*, volume 24, pages 109–165. Academic Press.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, and 2 others. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32.

Fuli Qiao and Mehrdad Mahdavi. 2024. Learn more, but bother less: Parameter efficient continual learning. In *Advances in Neural Information Processing Systems*, volume 37, pages 97476–97498. Curran Associates, Inc.

Chengwei Qin and Shafiq Joty. 2022. LFPT5: A unified framework for lifelong few-shot language learning based on prompt tuning of T5. In *Proceedings of the 10th International Conference on Learning Representations*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. 2023. Progressive prompts: Continual learning for language models. In *Proceedings of the 11th International Conference on Learning Representations*.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, Zifeng Wang, Sayna Ebrahimi, and Hao Wang. 2025. Continual learning of large language models: A comprehensive survey. *ACM Computing Surveys*, 58(5).

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. 2017. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32, Vancouver, Canada.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 7th International Conference on Learning Representations*, pages 1786–1805, New Orleans, USA.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. 2024. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(8):5362–5383.

Luping Wang, Sheng Chen, Linnan Jiang, Shu Pan, Runze Cai, Sen Yang, and Fei Yang. 2025. Parameter-efficient fine-tuning in large models: A survey of methodologies. *Computing Research Repository*, arXiv:2410.19878.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. 2023a. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10658–10671, Singapore.

Xiao Wang, Yuansen Zhang, Tianze Chen, Songyang Gao, Senjie Jin, Xianjun Yang, Zhiheng Xi, Rui Zheng, Yicheng Zou, Tao Gui, Qi Zhang, and Xuanjing Huang. 2023b. TRACE: A comprehensive benchmark for continual learning in large language models. *Computing Research Repository*, arXiv:2310.06762.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, and Xiaoqi Ren. 2022. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 139–149, New Orleans, LA, USA.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Tongtong Wu, Linhao Luo, Yuan-Fang Li, Shirui Pan, Thuy-Trang Vu, and Gholamreza Haffari. 2024. Continual learning for large language models: A survey. *Computing Research Repository*, arXiv:2402.01364.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2025. Qwen2.5 technical report. *Computing Research Repository*, arXiv:2412.15115.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 3987–3995.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Da-Wei Zhou, Qi-Wei Wang, Zhi-Hong Qi, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. 2024. Class-incremental learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9851–9873.

Appendix

This appendix provides additional details and analyses to support the main paper. We organize the appendix into three parts. First, we provide a theoretical justification for why Fisher information can be used to characterize task-dependent geometry, and further explain why the K-FAC principal subspaces of the activation and gradient covariance factors can reflect task similarity. Second, we present detailed experimental settings, including benchmark construction, implementation details, hyperparameters, and evaluation protocols. Third, we report additional experimental results and ablation studies to further analyze the behavior of FiUni under different configurations.

A Theoretical Justification

In this section, we provide a theoretical justification for why Fisher information can be used to identify different latent tasks, and why its K-FAC approximation allows us to use the principal subspaces of the activation and gradient covariance factors as a practical task-similarity signal. We first discuss the full Fisher information matrix, and then derive the connection under the K-FAC approximation.

A.1 Full Fisher Information as Task Geometry

Consider a pre-trained model with parameters θ_0 and a downstream task distribution $\mathcal{D}_t$. For a sample $z = (x, y) \sim \mathcal{D}_t$, let $\ell(z; \theta_0)$ denote the training loss, and define the task-induced gradient as

$$g_t(z) = \nabla_\theta \ell(z; \theta)\big|_{\theta=\theta_0}. \tag{14}$$

The Fisher information matrix (ichi Amari, 1998) induced by task $\mathcal{D}_t$ is

$$F_t = \mathbb{E}_{z\sim\mathcal{D}_t}\left[g_t(z) g_t(z)^\top\right]. \tag{15}$$

This matrix captures the second-order geometry of the loss around the pre-trained model. For a small parameter perturbation $\Delta\theta$, the local change of the task loss or predictive distribution can be approximated by a quadratic form:

$$\Delta\mathcal{L}_t \approx \frac{1}{2}\Delta\theta^\top F_t \Delta\theta. \tag{16}$$

Therefore, directions with large Fisher eigenvalues correspond to parameter directions to which task $\mathcal{D}_t$ is most sensitive. In other words, the dominant eigenspace of F_t describes the main update directions required by this task on the pre-trained model.

Let

$$Q_t = \mathrm{TopEig}(F_t, r) \tag{17}$$

be the subspace spanned by the top-r eigenvectors of F_t. For two tasks $\mathcal{D}_a$ and $\mathcal{D}_b$, their Fisher principal subspaces Q_a and Q_b reflect whether the two tasks require similar parameter update directions. A natural similarity measure is

$$s_F(a, b) = \frac{1}{r}\left\|Q_a^\top Q_b\right\|_F^2. \tag{18}$$

If the two tasks induce similar gradient distributions on the pre-trained model, then their Fisher matrices are close, and their principal subspaces are also close. If the two tasks require very different update directions, their Fisher principal subspaces tend to have smaller overlap.

This can be formalized using standard eigenspace perturbation results. Suppose the top-r eigenspace of F_a has eigengap γ_a, and F_b is a perturbation of F_a. Then the distance between their principal subspaces satisfies

$$\|\sin\Theta(Q_a, Q_b)\|_2 \le C\frac{\|F_a - F_b\|_2}{\gamma_a}, \tag{19}$$

where C is a constant and $\Theta(Q_a, Q_b)$ denotes the principal angles between the two subspaces. This indicates that if two tasks induce similar Fisher matrices, their principal subspaces remain aligned; if their Fisher matrices differ significantly, their principal subspaces become less aligned or more orthogonal.

Thus, the full Fisher information matrix provides a task-dependent geometric representation: its leading eigenspace captures the parameter-sensitive directions of a task, and the overlap between two Fisher principal subspaces reflects the similarity between the corresponding tasks.

A.2 K-FAC Approximation of the Layer-wise Fisher Matrix

Although the full Fisher information matrix provides a principled task geometry, it is prohibitively expensive to compute and store for large language models. Therefore, we use the K-FAC (Martens and Grosse, 2015) approximation to obtain a tractable layer-wise representation.

Consider a linear layer with weight matrix

$$W \in \mathbb{R}^{m\times n}. \tag{20}$$

Let $x \in \mathbb{R}^n$ be the input activation of this layer, and let $\delta \in \mathbb{R}^m$ be the back-propagated gradient with respect to the layer output. The gradient of the loss with respect to W is

$$\nabla_W \ell = \delta x^\top. \tag{21}$$

After vectorization, this gradient can be written as

$$\mathrm{vec}(\nabla_W \ell) = x \otimes \delta. \tag{22}$$

The layer-wise Fisher matrix is then

$$F_W = \mathbb{E}\left[\mathrm{vec}(\nabla_W \ell)\,\mathrm{vec}(\nabla_W \ell)^\top\right]. \tag{23}$$

K-FAC approximates this Fisher matrix by factorizing the second-order statistics of activations and output gradients:

$$F_W \approx \mathcal{G} \otimes \mathcal{A}, \tag{24}$$

where

$$\begin{aligned} \mathcal{A} &= \mathbb{E}\left[xx^\top\right], \\ \mathcal{G} &= \mathbb{E}\left[\delta\delta^\top\right]. \end{aligned} \tag{25}$$

Here, $\mathcal{A}$ captures the dominant directions in the input activation space, while $\mathcal{G}$ captures the dominant directions in the output-gradient space. Together, they approximate the layer-wise Fisher geometry induced by the current data.

A.3 Why the Principal Subspaces of $\mathcal{A}$ and $\mathcal{G}$ Reflect Task Similarity

We now explain why the top eigenvectors of $\mathcal{A}$ and $\mathcal{G}$ can be used to reflect task similarity under the K-FAC approximation.

Let

$$\begin{aligned} \mathcal{A} v_i &= \alpha_i v_i, \\ \mathcal{G} u_j &= \beta_j u_j. \end{aligned} \tag{26}$$

Then the Kronecker product $\mathcal{G} \otimes \mathcal{A}$ has eigenvectors of the form $u_j \otimes v_i$, with eigenvalues $\beta_j \alpha_i$:

$$(\mathcal{G} \otimes \mathcal{A})(u_j \otimes v_i) = \beta_j \alpha_i (u_j \otimes v_i). \tag{27}$$

Therefore, the dominant eigenspace of the K-FAC Fisher matrix is determined by the dominant eigenspaces of $\mathcal{G}$ and $\mathcal{A}$. This means that the leading eigenvectors of the gradient covariance factor and the activation covariance factor provide a compact approximation to the principal geometry of the layer-wise Fisher matrix.

For a data window or task $\mathcal{B}$, we define

$$\begin{aligned} U_{\mathcal{B}} &= \mathrm{TopEig}(\mathcal{G}_{\mathcal{B}}, r), \\ V_{\mathcal{B}} &= \mathrm{TopEig}(\mathcal{A}_{\mathcal{B}}, r). \end{aligned} \tag{28}$$

Here, $U_{\mathcal{B}}$ represents the dominant output-gradient directions, and $V_{\mathcal{B}}$ represents the dominant input-activation directions. Since the dominant directions of $\mathcal{G} \otimes \mathcal{A}$ are constructed from combinations of $U_{\mathcal{B}}$ and $V_{\mathcal{B}}$, the pair $(U_{\mathcal{B}}, V_{\mathcal{B}})$ can serve as a tractable representation of the Fisher principal geometry.

Now consider two data distributions or tasks $\mathcal{D}_a$ and $\mathcal{D}_b$. If they are similar, they tend to produce similar activation patterns and similar output-gradient patterns on the same pre-trained model. This implies that

$$\mathcal{A}_a \approx \mathcal{A}_b, \tag{29}$$

and

$$\mathcal{G}_a \approx \mathcal{G}_b. \tag{30}$$

Under eigengap conditions, the top eigenspaces of these matrices will also be close. Thus, V_a will be close to V_b, and U_a will be close to U_b. Conversely, if two tasks induce very different activation or gradient covariance structures, their top eigenspaces will have smaller overlap.

This provides the theoretical basis for using the principal subspaces of $\mathcal{A}$ and $\mathcal{G}$ to compare tasks. Instead of explicitly constructing the full Fisher eigenspace, FiUni measures the overlap between the two K-FAC factors:

$$s_U(\mathcal{B}_i, \mathcal{B}_j) = \frac{1}{r_{\mathrm{det}}} \left\| U_i^\top U_j \right\|_F^2, \tag{31}$$

$$s_V(\mathcal{B}_i, \mathcal{B}_j) = \frac{1}{r_{\mathrm{det}}} \left\| V_i^\top V_j \right\|_F^2. \tag{32}$$

The final similarity is

$$s(\mathcal{B}_i, \mathcal{B}_j) = \frac{s_U(\mathcal{B}_i, \mathcal{B}_j) + s_V(\mathcal{B}_i, \mathcal{B}_j)}{2}. \tag{33}$$

This similarity is a surrogate for the overlap between the dominant eigenspaces of the K-FAC Fisher matrices. A high value means that the two data windows share similar activation and gradient principal directions, and therefore induce similar Fisher geometry. A low value means that their sensitive update directions are more orthogonal, suggesting that they correspond to different latent tasks or learning phases.

A.4 Implication for Task-free Continual Learning

The above analysis shows that Fisher information contains task-dependent geometry at two levels. At the full-matrix level, the Fisher principal subspace captures the dominant parameter directions

to which a task is most sensitive. Under the K-FAC approximation, this geometry is represented by the principal subspaces of the activation covariance factor $\mathcal{A}$ and the gradient covariance factor $\mathcal{G}$.

This explains why the Fisher/K-FAC principal subspaces can be used for latent task identification in an online task-free stream. If an incoming batch belongs to an existing latent task, its Fisher principal subspace should have high similarity with historical subspaces. If it corresponds to a new or substantially different latent phase, its Fisher subspace should have low similarity to all historical subspaces. Intermediate similarity naturally corresponds to partial overlap, motivating the expansion operation in FiUni.

Therefore, Fisher geometry provides a unified signal for both detecting the latent task affiliation of each incoming batch and constructing the corresponding low-rank adaptation subspace.

Connection to subspace Methods. The above analysis also provides a possible explanation for why subspace methods, such as SeqLoRA and O-LoRA, can be effective in continual learning. We conjecture that, around a pre-trained model, different downstream tasks do not explore the entire parameter space uniformly, but instead tend to update along Fisher-sensitive directions induced by their task data. As a result, the effective update space used by each task may occupy only a low-dimensional subspace of the full parameter space. When different tasks correspond to distinct Fisher-sensitive directions, stochastic gradient descent may naturally push their updates toward partially separated or nearly orthogonal regions during training.

From this perspective, orthogonal constraints and subspace allocation mechanisms may not create task isolation from scratch; rather, they may reinforce the implicit geometric separation induced jointly by the pre-trained model and downstream tasks. Thus, the effectiveness of these methods may not solely come from the imposed orthogonality constraints, but may also rely on a task-related geometric structure that already exists near the pre-trained model: different tasks correspond to different dominant Fisher directions, while related tasks share partially overlapping directions. This perspective suggests that parameter isolation and knowledge reuse in continual learning can be understood as explicit ways of exploiting this pre-training-induced geometric structure.

B System-level Explanation of FiUni

Notably, the Fisher subspace similarity based detection mechanism is not tied to a specific LoRA adaptation design. Since it only relies on the geometric matching between the Fisher principal subspace of the current batch and historical subspaces, it can serve as a standalone upstream module for latent task identification and be plugged into other continual learning methods or task-expert selection scenarios. FiUni further combines this detection mechanism with Fisher-guided LoRA subspace construction to enable task-free continual parameter-efficient fine-tuning.

C Experimental Settings

C.1 Benchmarks

We evaluate FiUni on three continual learning benchmarks: **Standard CL Benchmark (SC)** (Zhang et al., 2015), **Long Sequence Benchmark (LS)** (Razdaibiedina et al., 2023), and **TRACE** (Wang et al., 2023b). These benchmarks cover task streams of different lengths, domains, and task formats. SC focuses on short text classification sequences, LS extends the setting to longer and more heterogeneous classification-oriented task streams by incorporating additional tasks from GLUE (Wang et al., 2019b) and SuperGLUE (Wang et al., 2019a), and TRACE further introduces diverse LLM-oriented tasks such as multilingual understanding, code completion, mathematical reasoning, and question answering. The benchmark composition and task orders are summarized in Tables 4, 5, and 6.

For the SC and LS benchmarks, we follow the same data sampling protocol as prior studies (Wang et al., 2023a; Liao et al., 2026). Specifically, we use the same task orders and the same sampled training and evaluation splits provided in their official code repositories, ensuring that both training and testing data are consistent with existing baselines for fair comparison.

In addition to hiding task boundaries, our setting differs from task-aware continual learning in how the data stream is constructed. We concatenate datasets from different tasks into a single continuous stream. As a result, around a manually defined task boundary, a training batch may contain samples from two substantially different datasets. This makes our task-free benchmark closer to realistic online scenarios and more challenging than the standard task-aware setting.

Dataset Name	Category	Task	Domain	Metric
Yelp	CL Benchmark	Sentiment Analysis	Yelp Reviews	Accuracy
Amazon	CL Benchmark	Sentiment Analysis	Amazon Reviews	Accuracy
DBpedia	CL Benchmark	Topic Classification	Wikipedia	Accuracy
Yahoo	CL Benchmark	Topic Classification	Yahoo Q&A	Accuracy
AG News	CL Benchmark	Topic Classification	News	Accuracy
MNLI	GLUE	Natural Language Inference	Various	Accuracy
QQP	GLUE	Paraphrase Detection	Quora	Accuracy
RTE	GLUE	Natural Language Inference	News, Wikipedia	Accuracy
SST-2	GLUE	Sentiment Analysis	Movie Reviews	Accuracy
WiC	SuperGLUE	Word Sense Disambiguation	Lexical Databases	Accuracy
CB	SuperGLUE	Natural Language Inference	Various	Accuracy
COPA	SuperGLUE	Question Answering	Blogs, Encyclopedia	Accuracy
BoolQA	SuperGLUE	Boolean Question Answering	Wikipedia	Accuracy
MultiRC	SuperGLUE	Question Answering	Various	Accuracy
IMDB	SuperGLUE	Sentiment Analysis	Movie Reviews	Accuracy

Table 4: Task details of the Long Sequence Benchmark (LS). The first five datasets correspond to the standard CL benchmark, while the remaining tasks are drawn from GLUE, SuperGLUE, and IMDB.

Dataset	Source	Avg. Len.	Metric	Language	#Data
Domain-specific					
ScienceQA	Science	210	Accuracy	English	5,000
FOMC	Finance	51	Accuracy	English	5,000
MeetingBank	Meeting	2853	ROUGE-L	English	5,000
Multi-lingual					
C-STANCE	Social Media	127	Accuracy	Chinese	5,000
20Minuten	News	382	SARI	German	5,000
Code Completion					
Py150	GitHub	422	Edit Similarity	Python	5,000
Mathematical Reasoning					
NumGLUE-cm	Math	32	Accuracy	English	5,000
NumGLUE-ds	Math	21	Accuracy	English	5,000

Table 5: Task details of TRACE. The benchmark covers domain-specific tasks, multilingual tasks, code completion, and mathematical reasoning.

C.2 Baselines

We compare FiUni with a broad set of continual learning and parameter-efficient fine-tuning baselines. These baselines cover sequential LoRA training, replay-based methods, regularization-based continual learning, prompt-based continual learning, and recent orthogonal or subspace-based continual PEFT methods.

SeqLoRA. SeqLoRA sequentially trains LoRA modules on the task stream without preserving previous task knowledge or allocating task specific subspaces. It serves as a simple continual PEFT baseline that directly applies LoRA to sequential learning.

SeqLoRAReplay. SeqLoRAReplay extends SeqLoRA with experience replay. During training on new tasks, it reuses a small number of stored examples from previous tasks to reduce forgetting.

IncLoRA. IncLoRA incrementally introduces new LoRA parameters for newly encountered tasks. Instead of using a single shared LoRA module, it expands adaptation parameters along the task sequence to provide additional capacity for new knowledge.

EWC (Kirkpatrick et al., 2017). Elastic Weight Consolidation is a regularization based continual learning method. It estimates parameter importance, typically using Fisher information, and penalizes changes to parameters important for previous tasks.

L2P (Wang et al., 2022). Learning to Prompt maintains a pool of learnable prompts and selects input relevant prompts during training. It adapts to new tasks by reusing and updating prompt knowledge from previous tasks.

LFPT5 (Qin and Joty, 2022). LFPT5 continuously

Benchmark	Order	Task Sequence
Standard CL Benchmark	1	DBpedia → Amazon → Yahoo → AG News
	2	DBpedia → Amazon → AG News → Yahoo
	3	Yahoo → Amazon → AG News → DBpedia
Long Sequence Benchmark	4	MNLI → CB → WiC → COPA → QQP → BoolQA → RTE → IMDB → Yelp → Amazon → SST-2 → DBpedia → AG News → MultiRC → Yahoo
	5	MultiRC → BoolQA → WiC → MNLI → CB → COPA → QQP → RTE → IMDB → SST-2 → DBpedia → AG News → Yelp → Amazon → Yahoo
	6	Yelp → Amazon → MNLI → CB → COPA → QQP → RTE → IMDB → SST-2 → DBpedia → AG News → Yahoo → MultiRC → BoolQA → WiC
TRACE	7	C-STANCE → FOMC → MeetingBank → Py150 → ScienceQA → NumGLUE-cm → NumGLUE-ds → 20Minuten

Table 6: Task orders used in our continual learning experiments. Orders 1–3 correspond to SC, Orders 4–6 correspond to LS, and Order 7 corresponds to TRACE.

trains soft prompts for T5 based continual learning. The learned prompts are used both for task solving and pseudo sample generation, which supports experience replay.

O-LoRA (Wang et al., 2023a). O-LoRA reduces task interference by enforcing orthogonality among the low rank update spaces of different tasks. In this way, task specific parameter updates are isolated in separate subspaces.

MIGU (Du et al., 2024). MIGU is a rehearsal free and task label free continual learning method. It exploits the magnitude distribution of outputs in linear layers and updates parameters with large output magnitudes to reduce gradient conflicts across tasks.

SpaRTA (Liao et al., 2026). SpaRTA is a continual PEFT method that manages task specific and shared adaptation components. It aims to reduce forgetting while preserving transfer across related tasks.

ELLA (Biswas et al., 2026). ELLA is a continual PEFT baseline that models task relationships in the adaptation space. It encourages knowledge sharing across related tasks while maintaining task specific components to reduce interference.

C.3 Implementation Details

We implement all experiments based on Transformers 4.57.6 (Wolf et al., 2020) and PyTorch 2.6.0 (Paszke et al., 2019). To ensure consistency across different methods, all baselines and FiUni are evaluated under the same software environment and hardware configuration. All experiments are conducted on a single NVIDIA RTX 6000 Ada Generation GPU. All models are initialized in bfloat16 (bf16) precision to reduce memory usage and computational overhead. Unless otherwise specified, all methods use the same backbone model, task orders, data splits, and evaluation protocol within each benchmark to ensure fair comparison.

For the SC and LS benchmarks, we concatenate all training data according to the predefined task orders and feed them into the model as a continuous batch stream. This setting simulates an online task-free continual learning scenario, where data arrive sequentially and the model must update itself based only on the current batch. During training, the model has no access to task boundaries or task IDs, and therefore cannot rely on explicit task-switching signals to create, select, or reset adaptation modules.

For the TRACE benchmark, different tasks involve different batch size requirements. Therefore, for data-format compatibility, we organize training batches task by task rather than directly mixing all samples into a single stream. However, this does not provide task-aware supervision to the model: task identities and explicit task-switching signals are still hidden during training. For each incoming batch, its latent task affiliation and the corresponding subspace management decision are determined adaptively by FiUni based on Fisher subspace similarity.

We summarize the hyperparameters used for different backbones and benchmarks in Tables 7 and 8. Here, r denotes the LoRA/FiLoRA subspace rank, r_{det} denotes the detection rank used for Fisher subspace matching, and Δr denotes the additional rank introduced during expansion. The thresholds τ_{low} and τ_{high} control the New, Expand, and Reuse decisions. We use $\mathcal{L}_{\text{adapt}}$ for the layers used in adapta-

Hyperparameter	T5-large			LLaMA-3.1-8B		
	SC	LS	TRACE	SC	LS	TRACE
Batch size			32			
Learning rate		1e-3			1e-4	
Epochs			1			
FiLoRA rank r		32			128	
Dropout	0	0.05	0	0.1	0.1	0
Adaptation layers $\mathcal{L}_{\text{adapt}}$			all			
Target modules $\mathcal{M}$		q,k,v,o			q,v	

Table 7: Training and adaptation hyperparameters for FiUni. Batch size denotes the effective batch size, computed as the raw batch size multiplied by the number of gradient accumulation steps.

Hyperparameter	T5-large			LLaMA-3.1-8B		
	SC	LS	TRACE	SC	LS	TRACE
Detection rank r_{det}		4			8	
Expansion rank Δr	0	4	4	8	8	8
τ_{low}		0.5			0.6	
τ_{high}			0.7			
Detection layers $\mathcal{L}_{\text{det}}$		23			31	
Expand cooldown C_{expand}	50	200	100	40	40	40
New cooldown C_{new}	30	10	100	10	10	10

Table 8: Detection and decision hyperparameters for FiUni.

tion, $\mathcal{L}_{\text{det}}$ for the layers used in Fisher-based detection, and $\mathcal{M}$ for the target modules. The cooldown steps C_{expand} and C_{new} prevent Expand and New operations from being triggered too frequently.

Threshold selection. The thresholds τ_{low} and τ_{high} in Table 8 are not finely tuned according to final test performance or manual task boundaries. Instead, they are coarsely set according to the natural scale of Fisher UV subspace similarity on each backbone, and are kept fixed across benchmarks and orders under the same backbone. We use 0-based layer indexing consistent with the model implementation. Therefore, layer 23 of T5-Large and layer 31 of LLaMA-3.1-8B correspond to the final Transformer block layers of the 24-layer and 32-layer models, respectively.

To provide an empirical basis for this setting, we conduct a task-similarity-based threshold estimation analysis. Specifically, we sample batches from different tasks on a frozen backbone, compute their Fisher UV principal-subspace similarity, and divide task pairs into three categories according to task relationships: unrelated, related, and same-task pairs. We then use the intersection between the unrelated and related distributions to estimate $\tau_{\text{low}}^{\text{est}}$, and the intersection between the related and same-task distributions to estimate $\tau_{\text{high}}^{\text{est}}$. This process is only used to analyze the natural scale of Fisher similarity and does not use FiUni's online decision results or final test performance.

As shown in Figure 4, different backbones exhibit different Fisher similarity scales. For T5-Large, the estimated thresholds are $\tau_{\text{low}}^{\text{est}} = 0.550$ and $\tau_{\text{high}}^{\text{est}} = 0.775$, while the actually used thresholds are $\tau_{\text{low}} = 0.500$ and $\tau_{\text{high}} = 0.700$. For LLaMA-3.1-8B, the estimated thresholds are $\tau_{\text{low}}^{\text{est}} = 0.649$ and $\tau_{\text{high}}^{\text{est}} = 0.831$, while the actually used thresholds are $\tau_{\text{low}} = 0.600$ and $\tau_{\text{high}} = 0.700$. The used thresholds lie in similarity regions consistent with the distribution intersections, but are slightly lower than the estimated intersections. This conservative setting improves online stability: a lower τ_{low} avoids assigning related batches with moderate drift too early to independent subspaces, while a lower τ_{high} helps maintain continuous reuse in stable phases and reduces unnecessary expansion.

D Additional Results

D.1 Ablation on Fisher Subspace Similarity Computation

We further analyze how the computation of Fisher subspace similarity s is affected by different design choices, including the detection rank r_{det}, the

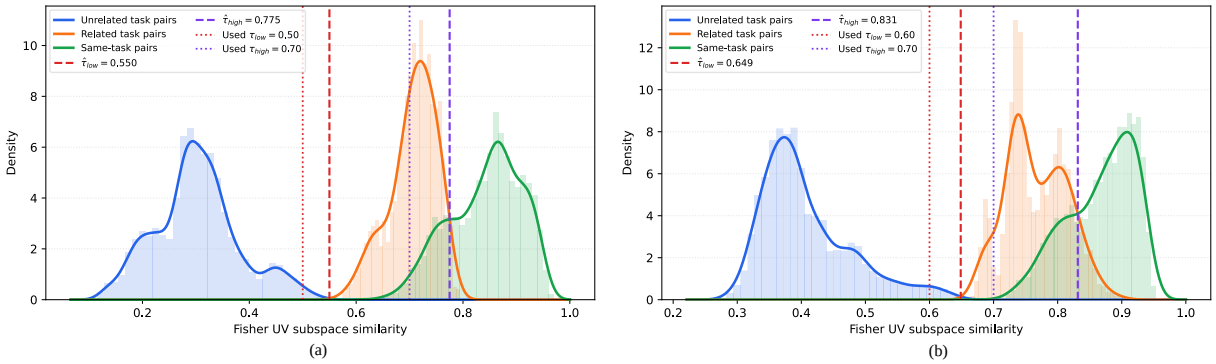

Figure 4: Task-similarity-based threshold estimation on different backbones. The left panel reports the results on T5-Large, and the right panel reports the results on LLaMA-3.1-8B. We divide task pairs into unrelated, related, and same-task pairs according to task relationships, and estimate $\tau_{\text{low}}^{\text{est}}$ and $\tau_{\text{high}}^{\text{est}}$ using the intersections between adjacent similarity distributions. The dotted vertical lines indicate the thresholds actually used in FiUni.

number of samples used for Fisher estimation, the selected modules, and the selected layers. Unless otherwise specified, the ablation is conducted on the LS benchmark using the same similarity computation protocol as in the main paper. Each figure reports both the task-level Fisher subspace similarity matrix and the scatter plot comparing within-task self-similarity with task-to-other similarity.

Effect of detection rank. Figure 5 shows the effect of using different detection ranks. When r_{det} is too small, the estimated Fisher subspace may not contain enough task-discriminative directions, leading to weaker separation between self-similarity and task-to-other similarity. As r_{det} increases, the similarity structure becomes more stable and better reflects task-level relationships. However, using an overly large rank may also introduce less informative directions, making the similarity signal less concentrated. This motivates the use of a moderate detection rank in FiUni.

Effect of Fisher estimation samples. Figure 6 studies the influence of the number of samples used for estimating the Fisher/K-FAC factors. With very few samples, the estimated Fisher statistics are noisier, and the resulting similarity signal becomes less stable. Increasing the number of samples improves the consistency of within-task self-similarity and makes the separation from unrelated tasks clearer. These results show that Fisher subspace similarity can be estimated from a small number of downstream samples, while a sufficient sample size is still helpful for robust task discrimination.

Resampling stability under 32 samples. The 32-sample setting is not obtained through tuning, but follows the continual training protocol where FiUni and the baselines use the same batch size under the same data split. Therefore, we use one standard training batch to estimate the Fisher principal subspace of the current batch, making the detection granularity consistent with the online training process.

As shown in Table 9, the task-similarity matrices remain highly consistent under independent 32-sample resamplings, with an average matrix correlation of 0.9943. The numerical variation of individual task-pair similarities is also small. These results indicate that 32 samples can provide stable Fisher-subspace similarity estimates while remaining consistent with the online training protocol.

To quantify the effect of random sampling noise, we conduct a repeated 32-sample resampling stability experiment on T5-base under the LS benchmark. We use 10 different random seeds to resample 32 examples from each task and reconstruct the complete task-similarity matrix. Matrix Corr. measures the overall pattern consistency between similarity matrices obtained from different resamplings, while Mean/Max Sim. Std. measures the numerical variation of each fixed task-pair similarity across resamplings.

Effect of module selection. Figure 7 compares different choices of Transformer modules for Fisher subspace estimation. The results indicate that the similarity pattern is relatively robust across different module selections. In particular, attention-related modules already provide meaningful task-discriminative signals, suggesting that FiUni does not require estimating Fisher statistics on all modules to obtain useful task geometry. This supports

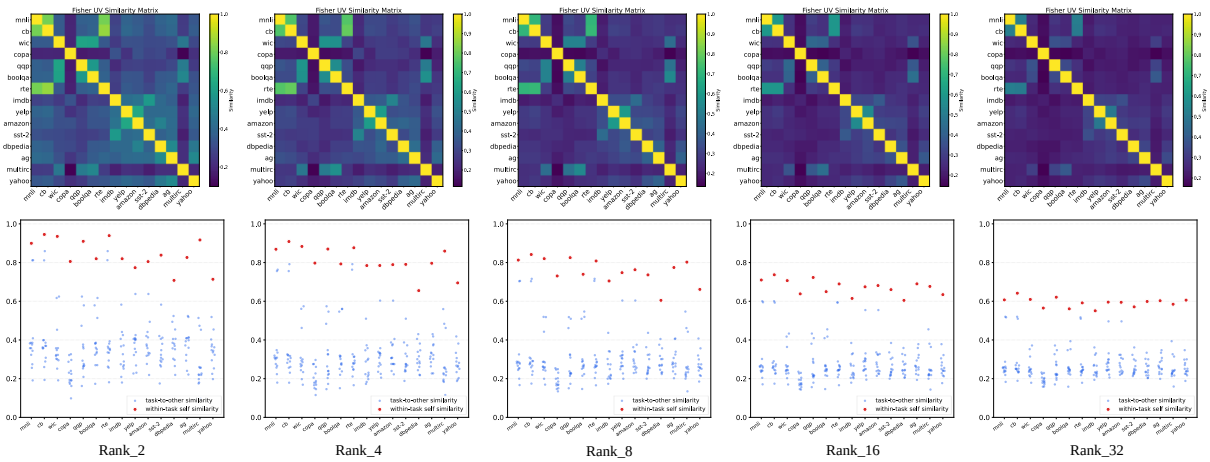

Figure 5: Ablation on the detection rank r_{det} for Fisher subspace similarity computation. A moderate rank provides a stable separation between within-task self-similarity and task-to-other similarity.

Sample Size	Matrix Corr. ↑	Corr. Std. ↓	Mean Sim. Std. ↓	Max Sim. Std. ↓
32	0.9943	0.0016	0.0073	0.0194

Table 9: Repeated 32-sample resampling stability analysis on T5-base under the LS benchmark.

our design of performing detection only on selected modules to reduce computation.

Effect of layer selection. We further evaluate whether Fisher subspace similarity depends strongly on the choice of layers. In addition to the representative layer cases, we compute Fisher UV similarity distributions over all 12 Transformer block layers of T5-base. For each layer, we compute pairwise Fisher UV similarities between task pairs and compare the resulting distributions across layers.

As shown in Figure 9, the similarity distributions remain broadly consistent across different T5 block layers, and the layer-wise mean similarities stay in a narrow range. This indicates that the Fisher UV similarity signal is not tied to a specific selected layer. Therefore, using the final Transformer block layer for detection provides an efficient approximation while preserving the overall task-similarity structure.

Overall, these ablations show that Fisher subspace similarity is a robust signal for capturing task-level geometry. Although the quality of the signal is affected by the number of estimation samples and the detection rank, it remains relatively stable across different module and layer selections. This supports FiUni's practical design of using a moderate detection rank, a small number of Fisher estimation samples, and only selected modules/layers for efficient online task-free detection. Meanwhile, although the overall similarity values decrease under extremely few-shot estimation and the separation between related and unrelated tasks becomes less pronounced, the within-task similarity still remains the highest for most tasks. This suggests that Fisher subspace similarity can still preserve a certain degree of task-discriminative signal in low-sample regimes, providing a practical basis for sample-efficient batch-level detection in downstream continual adaptation.

D.2 Training Dynamics and Decision Trajectory

We further analyze the online behavior of FiUni by jointly examining the training loss and the corresponding batch-level decision trajectory. Figures 10 and 11 present the results of T5-large on the LS and SC benchmarks, respectively. In each figure, the upper panel shows the training loss along the sequential data stream, and the lower panel shows the corresponding Reuse, Expand, and New decisions.

Across both benchmarks, the decision trajectory follows the training dynamics in a meaningful yet nontrivial manner. When the incoming data become substantially less compatible with previously accumulated knowledge, the loss often rises and FiUni correspondingly activates New or Expand to introduce additional subspace capacity. By contrast, when the stream remains relatively stable,

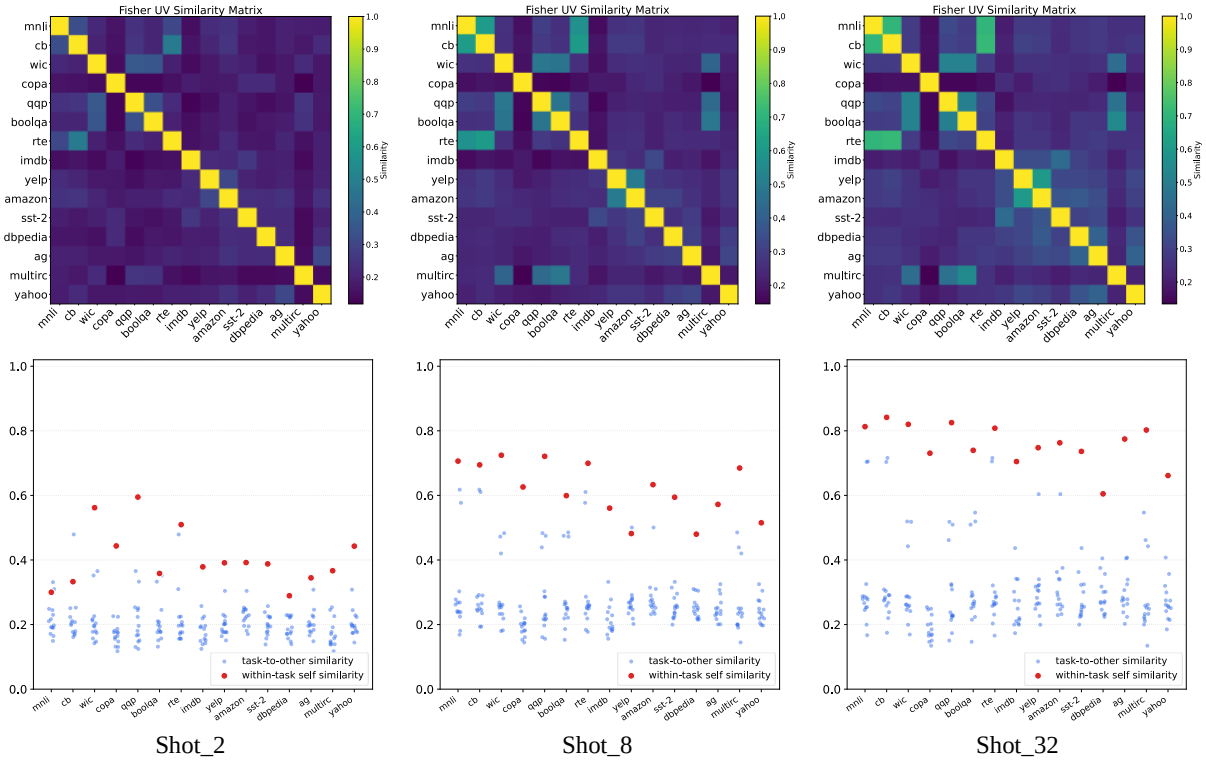

Figure 6: Ablation on the number of samples used for Fisher estimation. More samples lead to more stable Fisher subspaces and clearer separation between within-task and cross-task similarities.

the decisions are dominated by Reuse, meaning that the current batches can be effectively absorbed by existing Fisher subspaces. Such a pattern is particularly visible on the LS benchmark, where several major shifts in the stream are accompanied by noticeable changes in the loss curve and the appearance of newly allocated subspaces.

At the same time, FiUni is not merely reacting to abrupt loss changes. Compared with methods that rely on sudden loss variation for task-boundary detection, FiUni operates at a finer batch-level granularity through Fisher subspace matching. Consequently, its decision process is able to capture latent task transitions even when the loss curve does not exhibit a pronounced spike. For example, in the SC benchmark, the transitions from Amazon to Yahoo and from Yahoo to AG do not lead to particularly sharp loss peaks, yet FiUni still introduces new subspaces at the corresponding phases. A similar phenomenon appears in the LS benchmark, such as the transitions from Amazon to SST-2 and from AG to MultiRC, where the loss changes only mildly while the decision trajectory still detects the underlying shift in task geometry.

Another noteworthy characteristic is that FiUni does not simply replicate manually defined task boundaries. Some transitions are handled through Reuse or Expand rather than immediately opening a new subspace, especially when adjacent tasks remain geometrically related. Likewise, within a single human-defined task, FiUni may occasionally expand an existing subspace when the incoming batches reveal additional internal variation. This behavior is consistent with our notion of latent tasks: the model tracks its own perceived learning phases rather than mechanically following dataset-level segmentation.

In addition, in the SC benchmark, we observe several Expand decisions within the DBpedia stage. Although DBpedia is treated as a single dataset-level task, it covers a wide range of entity categories and topic types, which can induce noticeable batch-level variation in Fisher geometry. These batches may not be fully explained by the initial DBpedia subspace, but they remain related enough to avoid opening entirely new subspaces. FiUni therefore expands the existing subspace to provide additional capacity for intra-task variation while preserving knowledge reuse. The different behavior of DBpedia in SC and LS further reflects the context-dependent nature of FiUni's decisions. Rather than assigning a fixed decision pattern to

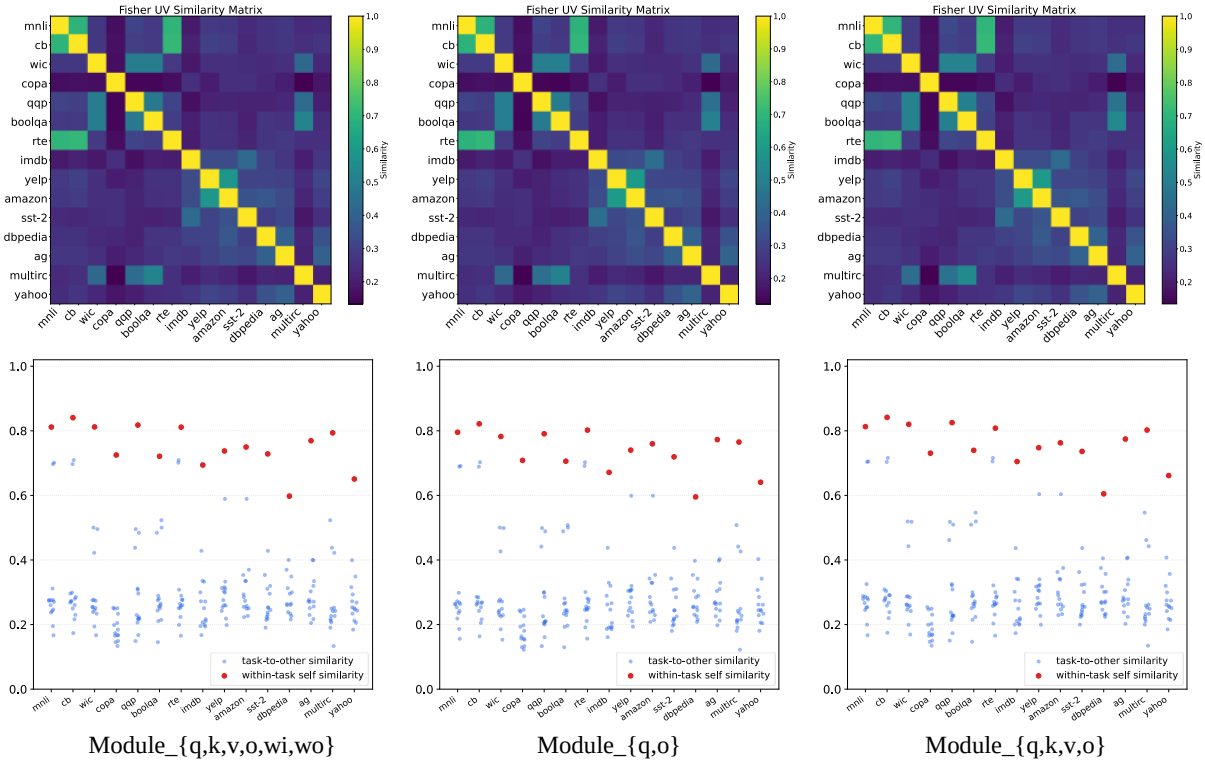

Figure 7: Ablation on module selection for Fisher subspace similarity computation. Different module choices preserve similar task-level structures, showing that the similarity signal is not tied to a single module configuration.

each dataset, FiUni compares every incoming batch with the current historical Fisher subspace pool. In SC, DBpedia appears when the historical pool is still small, making its intra-task variation more likely to fall into the intermediate-similarity region and trigger Expand. In LS, however, DBpedia is encountered after many previous tasks have already populated the subspace pool, so some DBpedia batches can be better explained by existing related subspaces and are thus handled through Reuse. This behavior further supports that FiUni tracks model-perceived latent learning phases rather than simply following dataset names or manually defined task boundaries.

Taken together, the loss-decision analysis provides an intuitive view of how FiUni manages subspaces throughout online continual adaptation. Large shifts in the stream are often accompanied by subspace creation or expansion, while stable periods are mainly handled through knowledge reuse. More importantly, the decision mechanism is not limited to coarse signals derived from loss fluctuations, but can also resolve finer latent transitions at the batch level, which is especially valuable in realistic task-free settings where explicit task boundaries are unavailable.

D.3 Dynamic Trainable Parameter Analysis

We further analyze the number of trainable parameters from a dynamic perspective on the LS benchmark with T5-large. Different from task-aware methods that typically allocate or optimize a fixed amount of adaptation parameters for each task, FiUni adjusts its trainable parameter budget according to the online decision trajectory. Specifically, a Reuse decision does not introduce additional subspace parameters, while Expand only increases the capacity of an existing subspace by a small amount. A New decision allocates a new LoRA subspace only when the incoming batch is geometrically distinct from all historical subspaces. Therefore, the trainable parameter count of FiUni grows adaptively with the data stream rather than linearly following the number of manually defined tasks.

For comparison, according to the hyperparameter settings reported in prior work (Wang et al., 2023a; Liao et al., 2026), we compute the accumulated trainable parameters of O-LoRA and SpaRTA on T5-large, while for FiUni we directly count the dynamically allocated trainable parameters after training on Order 4. The comparison is summarized in Table 10.

The results show that FiUni uses substantially

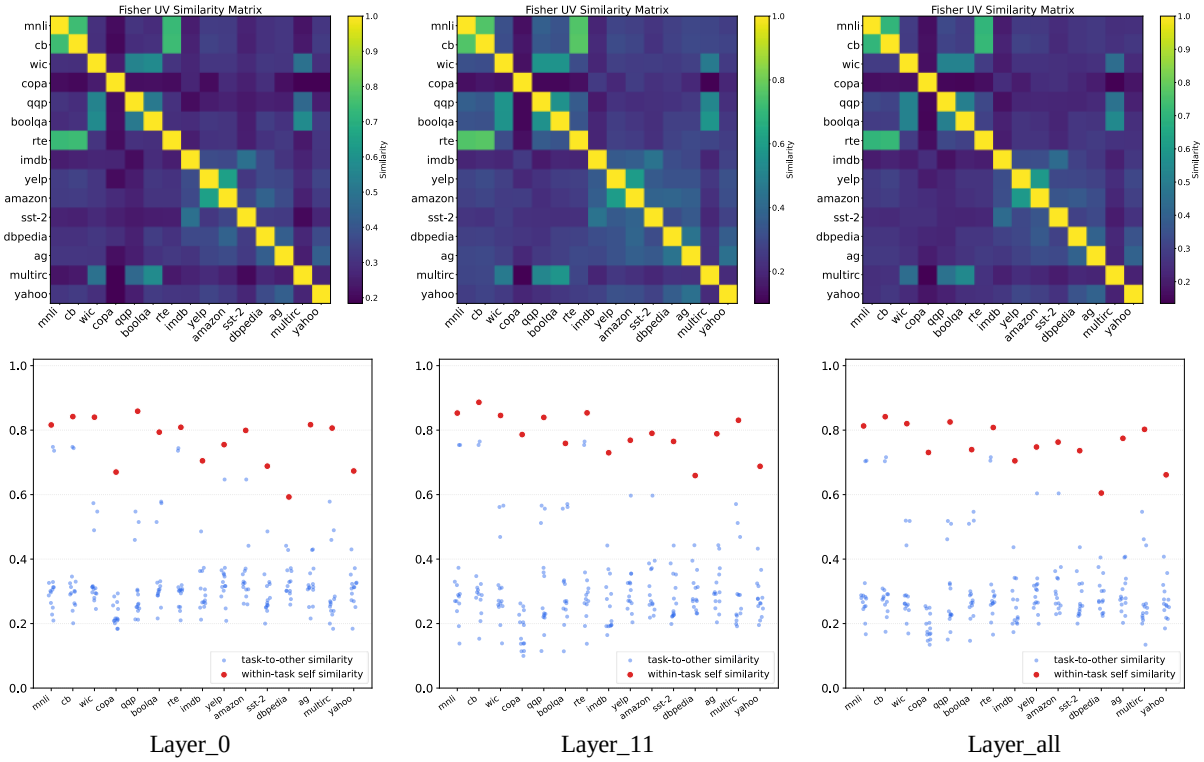

Figure 8: Ablation on layer selection for Fisher subspace similarity computation. The resulting similarity structures remain broadly consistent across different layer choices, supporting efficient detection with selected layers.

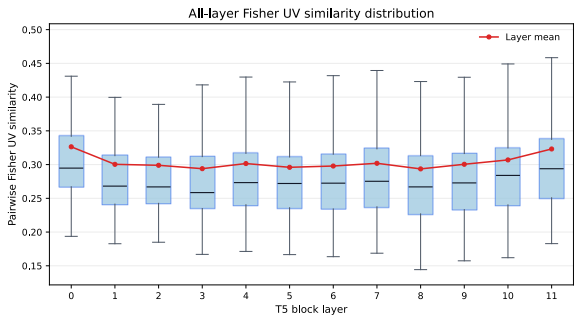

Figure 9: All-layer Fisher UV similarity distribution on T5-base. Each box shows the pairwise Fisher UV similarity distribution of one Transformer block layer, and the red curve indicates the layer-wise mean. The similarity distributions remain broadly consistent across layers, supporting the stability of Fisher UV similarity under different layer selections.

Metric	O-LoRA	SpaRTA	FiUni (ours)
Trainable Params	35.39M	44.24M	3.62M

Table 10: Trainable parameter comparison on the LS benchmark. For FiUni, we report the dynamic trainable parameter count obtained on Order 4.

fewer trainable parameters than the fixed task-level allocation adopted by strong task-aware baselines. Since FiUni only trains the core matrix within each Fisher-guided LoRA subspace, the trainable parameter count is substantially reduced at the subspace level. On top of this, the adaptive mechanism further helps control parameter growth: related batches can share existing subspaces, moderately shifted batches only expand the most relevant subspace, and new subspaces are created only when necessary. As a result, FiUni avoids assigning a full set of adaptation parameters to every dataset-level task, while still preserving enough flexibility to handle distributional changes in the online stream. This adaptive parameter allocation is especially beneficial for long-sequence continual learning, where blindly adding a complete adapter for each task would lead to rapid parameter growth.

D.4 End-to-end Mechanism Ablations

We further conduct end-to-end mechanism ablations to isolate the contribution of each major decision component in FiUni. The experiments are conducted on T5-Large under LS Order 5, while keeping the data stream, training setup, thresholds, Fisher estimation configuration, and other hyperparameters unchanged. We choose Order 5 because the full FiUni naturally contains a richer combina-

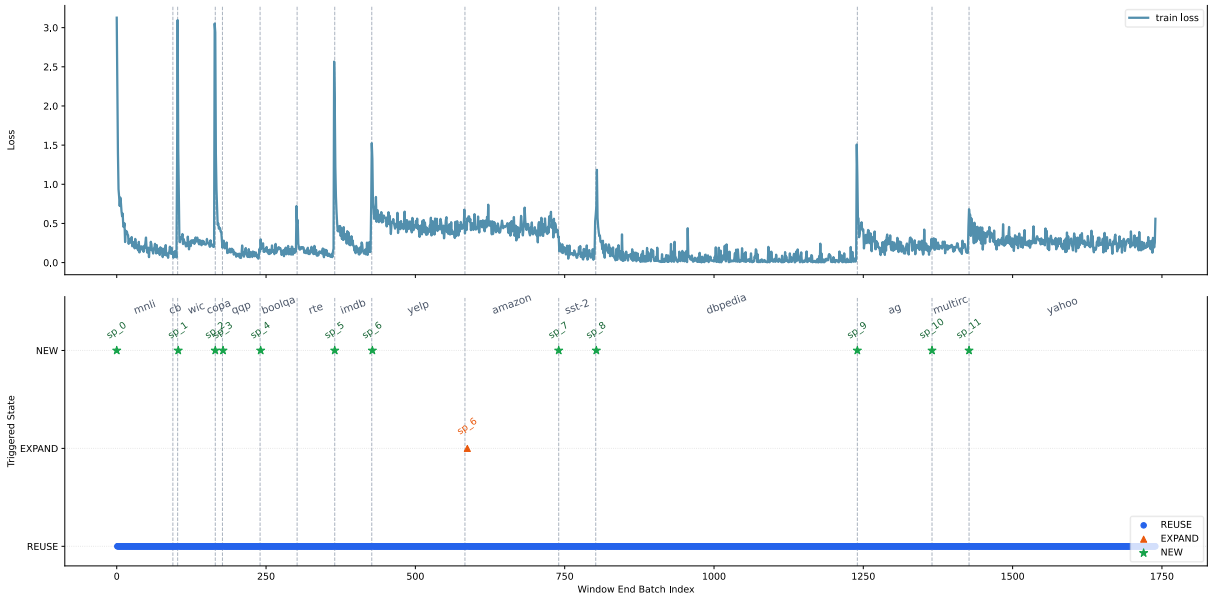

Figure 10: Training loss and FiUni decision trajectory on the LS benchmark with T5-large. The upper panel shows the training loss along the online data stream, and the lower panel shows the corresponding batch-level **Reuse**, **Expand**, and **New** decisions.

Variant	LS Order 5 ↑	Drop
Full FiUni	0.6700	–
w/o EXPAND	0.6410	−0.0290
w/o NEW	0.5817	−0.0883
Single-Subspace	0.6275	−0.0425
Oracle-Random	0.6010	−0.0690

Table 11: End-to-end mechanism ablations on T5-Large under LS Order 5.

tion of REUSE, EXPAND, and NEW decisions in this stream, making it suitable for evaluating the overall three-state decision mechanism. All results are averaged over three random seeds.

We construct four variants. *w/o Expand* replaces each original EXPAND decision with REUSE of the most similar historical subspace. *w/o New* replaces each original NEW decision with EXPAND of the nearest historical subspace. *Single-Subspace* constructs only one Fisher-guided subspace at the initial stage and reuses it throughout the stream. *Oracle-Random* uses true task boundaries as oracle information and allocates a new random subspace for each new task, which tests whether the performance gain merely comes from boundary timing or additional random low-rank capacity.

As shown in Table 11, all ablation variants underperform the full FiUni. Removing EXPAND decreases the performance from 0.6700 to 0.6410, indicating that directly reusing the nearest historical subspace is insufficient for related but moderately shifted batches. Removing NEW leads to the largest drop, showing that low-similarity phases require independent subspaces for isolation. Single-Subspace also performs worse than FiUni, suggesting that a single fixed Fisher-guided basis cannot explain the full gain. Finally, Oracle-Random remains clearly below FiUni even though it uses true task boundaries, indicating that FiUni's improvement comes from Fisher-guided subspace construction and online knowledge allocation, rather than boundary timing or random low-rank capacity alone.

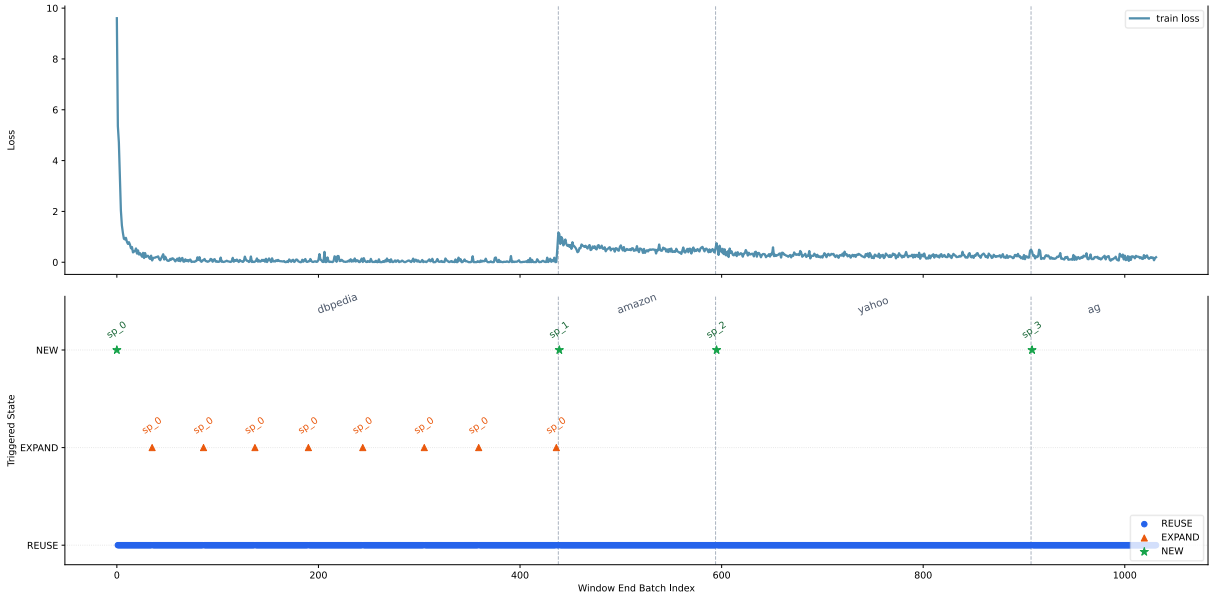

Figure 11: Training loss and FiUni decision trajectory on the SC benchmark with T5-large. Even when some task transitions do not produce obvious loss spikes, FiUni can still identify the corresponding latent shifts through Fisher subspace matching and trigger appropriate batch-level decisions.