

ReProF: Residual Prototype Fusion for Continual LLM Attribution

Yujie Wang¹, Zhendong Mao², Quan Wang^{1*}

¹MOE Key Laboratory of Trustworthy Distributed Computing and Service,
Beijing University of Posts and Telecommunications, Beijing, China

²University of Science and Technology of China, Hefei, China

{wangyujie01, wangquan}@bupt.edu.cn, zdmao@ustc.edu.cn,

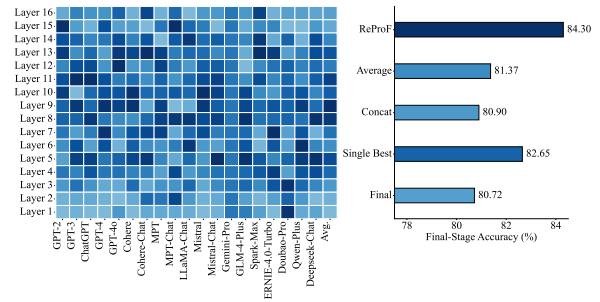
Abstract

Continual LLM attribution aims to identify the source of generated text amid the ongoing emergence of new models. While existing training-free, prototype-based methods excel at this task, they rely solely on the final-layer features of a frozen extractor, overlooking rich hierarchical stylistic cues. Moreover, naive multi-layer fusion yields suboptimal results. To address these limitations, we propose ReProF (Residual Prototype Fusion), a training-free framework that partitions the feature extractor’s hierarchy into layer blocks and derives depth-aware analytic prototypes via shallow-to-deep residual fitting: deeper prototypes model the residuals of shallower ones. By fusing predictions across these blocks, ReProF captures multi-granular stylistic cues without requiring parameter tuning or data rehearsal, thereby intrinsically eliminating catastrophic forgetting. Extensive experiments across 19 LLMs and 8 domains demonstrate that ReProF establishes a new state-of-the-art, consistently outperforming existing approaches by a wide margin. Code is available at: <https://github.com/wyj-hb/ReProF>.

1 Introduction

While the rapid advancement of large language models (LLMs) brings remarkable capabilities, it also raises serious risks like phishing (Roy et al., 2024), disinformation (Vykopal et al., 2024), and plagiarism (Lee et al., 2025). Consequently, **LLM attribution**, the task of identifying which LLM generated a given text, has become critical to responsible AI, serving as an essential tool to trace malicious content, enforce model accountability, and ensure ecosystem safety (Huang et al., 2025).

Existing approaches typically frame LLM attribution as a static multi-class classification problem, where all candidate LLMs are known at training time and a single fixed classifier is trained on their



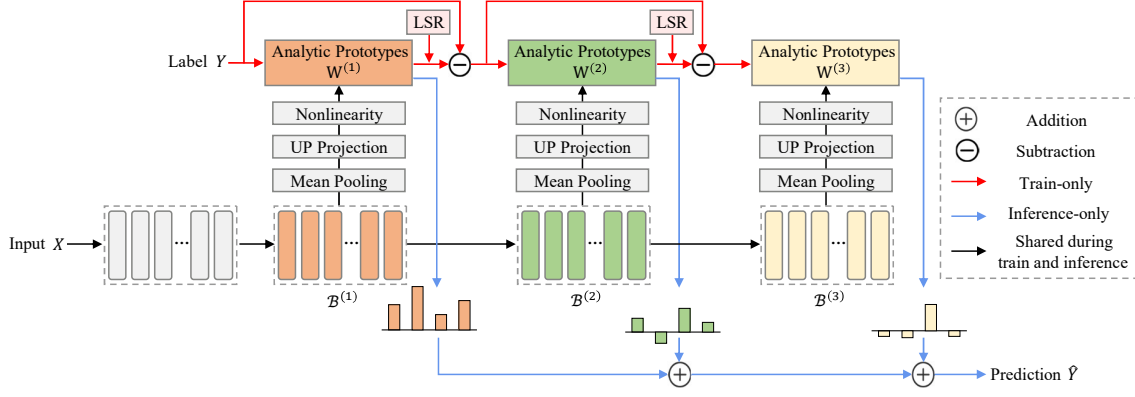


Figure 2: Overall architecture of ReProF. A frozen feature extractor is partitioned into contiguous, non-overlapping layer blocks (dashed boxes); the K deepest blocks (here $K = 3$) are used for prototype construction. During training, inputs are processed block-wise (black path), followed by shallow-to-deep residual prototype learning (red path), where LSR means label space restriction. At test time, block-wise features are aggregated cumulatively for prediction (blue path).

provides analytic prototypes for all encountered LLMs without historical samples. Being entirely training- and rehearsal-free, these methods inherently eliminate catastrophic forgetting, making them exceptionally well-suited for dynamic settings.

Despite their success, existing prototype-based methods share a critical bottleneck: they derive prototypes exclusively from the final layer of the frozen feature extractor. However, pre-trained language models often encode complementary linguistic properties across their internal hierarchy, ranging from surface lexical patterns in shallow layers to high-order syntactic structures in deeper ones (Jawahar et al., 2019; Tenney et al., 2019). Since the extractor cannot be fine-tuned in this training-free paradigm, relying on any single layer permanently discards discriminative stylistic cues embedded elsewhere. Our empirical analysis on the state-of-the-art method RanPAC (McDonnell et al., 2023) confirms this bottleneck: the optimal layer for attribution varies substantially across different LLMs, with the final layer rarely yielding the best performance (Fig. 1(a)). Furthermore, this cross-model heterogeneity undermines naive static fusion strategies (e.g., averaging or concatenation), which combine layers indiscriminately and mix beneficial signals with noise, thereby degrading overall effectiveness (Fig. 1(b)). These findings motivate a more adaptive hierarchical fusion strategy.

To this end, we devise **ReProF** (**Residual Prototype Fusion**), a training-free framework that leverages the full hierarchy of a frozen feature extractor to construct and fuse complementary prototypes for continual LLM attribution (Fig. 2). Re-

ProF first partitions the extractor’s hierarchy into contiguous layer blocks characterized by high intra-block similarity and sharp inter-block divergence, from which it selects a subset of informative deep blocks. Within each selected block, a closed-form classifier is constructed via ridge regression, with its weight vectors serving as analytic prototypes for the source LLMs. These prototypes are then fused via shallow-to-deep residual refinement: each deeper classifier fits the residuals left by preceding shallower blocks, accounting for previously unexplained attribution signals. This progressive refinement enables adaptive, data-driven fusion of hierarchical stylistic cues. Crucially, as all prototypes are derived analytically without gradient descent, the entire framework remains training-free and inherently immune to catastrophic forgetting.

We evaluate ReProF on standard benchmarks (Li and Wang, 2025) where new models from diverse LLM families are sequentially introduced. ReProF consistently outperforms the state of the art by substantial margins across eight distinct domains, validating its broad efficacy across diverse domains. Ablation studies further confirm that both hierarchical block partitioning and residual fusion are essential to ReProF’s success.

Our contributions are threefold. (1) We uncover a critical bottleneck in existing prototype-based methods, empirically proving that relying solely on the final layer discards vital hierarchical stylistic cues. (2) We propose ReProF, a training-free framework that partitions the feature extractor’s hierarchy into complementary blocks and fuses analytic prototypes via shallow-to-deep residual refinement. (3)

We establish a new state of the art, delivering consistent performance gains over existing methods across 19 source LLMs and 8 domains.

2 Preliminaries

This section formally defines the task and reviews the analytic prototype paradigm that underpins current training-free approaches.

2.1 Continual LLM Attribution

Continual LLM attribution is to identify the source LLM of a given text in open-world settings (Li and Wang, 2025). Framed as a class-incremental learning problem, the task proceeds across T sequential stages, corresponding to a data stream $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_T\}$. Each stage $\mathcal{D}_t = \{(x_j, y_j)\}_{j=1}^{N_t}$ introduces a disjoint label set of text generators $\mathcal{Y}_t$, where x_j is the text generated by source $y_j \in \mathcal{Y}_t$. The human author is included in the initial stage $\mathcal{Y}_1$. All generators are ordered chronologically by release dates, ensuring that for any $t < t'$, all sources in $\mathcal{Y}_t$ precede those in $\mathcal{Y}_{t'}$. At stage t , the system only has access to the current dataset $\mathcal{D}_t$, yet it must attribute texts to any source encountered thus far from the cumulative label space $\mathcal{Y}_{\leq t} = \bigcup_{s=1}^t \mathcal{Y}_s$, inherently challenging the model to prevent catastrophic forgetting without historical data access.

2.2 Analytic Prototype Paradigm

In continual LLM attribution, a prominent training-free paradigm utilizes the closed-form weight vectors of a linear classifier as analytic prototypes for classification. RanPAC (McDonnell et al., 2023) is a canonical implementation of this approach.

Specifically, at stage t , given the dataset $\mathcal{D}_t$, RanPAC employs a frozen language model to extract a base feature vector $f(x_j) \in \mathbb{R}^d$ for each text, typically computed as the average of its last-layer hidden states. This vector is then projected into a higher-dimensional space via a fixed random matrix $\mathbf{R} \in \mathbb{R}^{d \times D}$ ($D > d$) and an activation $\phi(\cdot)$:

$$\mathbf{z}_j = \phi(\mathbf{R}^\top f(x_j)) \in \mathbb{R}^D. \quad (1)$$

Let $\mathbf{Z}_t \in \mathbb{R}^{N_t \times D}$ and $\mathbf{Y}_t \in \mathbb{R}^{N_t \times |\mathcal{Y}_t|}$ denote the matrices of projected features and corresponding one-hot label encodings for stage t .

To discriminate among all observed sources up to stage t , RanPAC optimizes a ridge regression objective over the cumulative data:

$$\min_{\mathbf{W}_t} \|\mathbf{Z}_{\leq t} \mathbf{W}_t - \mathbf{Y}_{\leq t}\|_F^2 + \lambda \|\mathbf{W}_t\|_F^2, \quad (2)$$

where $\mathbf{Z}_{\leq t} = [\mathbf{Z}_1^\top, \dots, \mathbf{Z}_t^\top]^\top$ stacks all historical features, $\mathbf{Y}_{\leq t}$ concatenates the zero-padded one-hot matrices $\{\mathbf{Y}_s\}_{s=1}^t$ aligned with $\mathcal{Y}_{\leq t}$, and $\lambda > 0$ is a regularization parameter. Instead of explicitly storing historical matrices, the closed-form solution $\mathbf{W}_t \in \mathbb{R}^{D \times |\mathcal{Y}_{\leq t}|}$ can be computed without accessing historical data $\mathcal{D}_1, \dots, \mathcal{D}_{t-1}$ by maintaining two incrementally updated sufficient statistics:

$$\mathbf{G}_t = \mathbf{G}_{t-1} + \mathbf{Z}_t^\top \mathbf{Z}_t \in \mathbb{R}^{D \times D}, \quad (3)$$

$$\mathbf{C}_t = [\mathbf{C}_{t-1}, \mathbf{Z}_t^\top \mathbf{Y}_t] \in \mathbb{R}^{D \times |\mathcal{Y}_{\leq t}|}, \quad (4)$$

where $\mathbf{G}_0 = \mathbf{0}$ and $\mathbf{C}_0 = []$. Since $\mathbf{G}_t = \mathbf{Z}_{\leq t}^\top \mathbf{Z}_{\leq t}$ and $\mathbf{C}_t = \mathbf{Z}_{\leq t}^\top \mathbf{Y}_{\leq t}$, the classifier weights are updated rehearsal-free via:

$$\mathbf{W}_t = (\mathbf{G}_t + \lambda \mathbf{I})^{-1} \mathbf{C}_t, \quad (5)$$

where the columns serve as the analytic prototypes for all source LLMs observed up to stage t .

At inference, the prediction for a test sample x_{test} with projected feature $\mathbf{z}_{\text{test}} = \phi(\mathbf{R}^\top f(x_{\text{test}}))$ is given by matching against the prototypes:

$$y_{\text{pred}} = \arg \max_{y \in \mathcal{Y}_{\leq t}} \mathbf{z}_{\text{test}}^\top \mathbf{w}_y, \quad (6)$$

where $\mathbf{w}_y$ is the y -th column of $\mathbf{W}_t$.

By eliminating gradient-based optimization, this paradigm inherently avoids catastrophic forgetting and has inspired various extensions (Ahrens et al., 2023; Peng et al., 2025; Zhao et al., 2024; Li and Wang, 2025). However, these methods either constrain themselves to final-layer representations or employ naive static fusion, failing to harvest the rich discriminative stylistic cues distributed across the extractor’s hierarchy.

3 Our Method

This section introduces Residual Prototype Fusion (ReProF), a training-free framework that tackles continual LLM attribution by residually fusing hierarchical prototypes to capture multi-granular stylistic cues. As illustrated in Fig. 2, ReProF operates in two steps: (1) Layer Block Partitioning, which segments the hidden layers of the frozen language model into contiguous blocks; (2) Residual Prototype Construction & Fusion, which builds analytic prototypes across these blocks and aggregates them via shallow-to-deep residual refinement.

3.1 Layer Block Partitioning

Given a language model of depth L , ReProF partitions its layers into contiguous, non-overlapping

blocks based on representational similarity. We employ Singular Vector Canonical Correlation Analysis (SVCCA) (Raghu et al., 2017) as our primary metric. Crucially, this partitioning is calculated once using data from the first stage $\mathcal{D}_1$ and remains fixed throughout all subsequent stages. Comparisons regarding alternative metrics and data stages for partitioning are provided in Appendix E.

Specifically, for each text in $\mathcal{D}_1$ (N_1 samples), we extract the hidden states at layer $\ell \in \{1, \dots, L\}$ and apply mean pooling over the token dimension to construct the layer-wise representation matrix $\mathbf{H}^{(\ell)} \in \mathbb{R}^{N_1 \times d}$. We then segment these layers into distinct blocks by traversing greedily from deep to shallow layers. Let $\mathcal{B}_{\text{curr}}$ denote the block currently under construction, whose centroid representation is the arithmetic mean of its constituent layers:

$$\mathbf{H}_{\text{curr}} = \frac{1}{|\mathcal{B}_{\text{curr}}|} \sum_{\ell \in \mathcal{B}_{\text{curr}}} \mathbf{H}^{(\ell)}. \quad (7)$$

For the adjacent shallower candidate layer ℓ_{new} , we evaluate its similarity to the current block centroid via SVCCA:

$$s = \text{SVCCA}(\mathbf{H}^{(\ell_{\text{new}})}, \mathbf{H}_{\text{curr}}) \in [0, 1]. \quad (8)$$

If $s \geq \tau$, where $\tau \in (0, 1)$ is a pre-specified threshold, ℓ_{new} is merged into $\mathcal{B}_{\text{curr}}$ and the centroid is updated. Otherwise, $\mathcal{B}_{\text{curr}}$ is finalized and a new block is initialized containing only ℓ_{new} . The greedy strategy yields blocks with high intra-block similarity and distinct inter-block boundaries. The whole procedure is detailed in Algorithm 1 in Appendix A.

3.2 Residual Prototype Construction & Fusion

Given the partitioned blocks, ReProF selects the K deepest ones to construct block-specific analytic prototypes sequentially via shallow-to-deep residual fitting. This pipeline proceeds in three phases: feature extraction, residual learning, and cumulative inference.

Block-wise Feature Extraction Let $\{\mathcal{B}^{(1)}, \dots, \mathcal{B}^{(K)}\}$ denote the K selected blocks ordered from shallow to deep. At stage t , for each input text $x \in \mathcal{D}_t$, we mean-pool its token-level hidden states into the layer-wise representation $h^{(\ell)}(x) \in \mathbb{R}^d$, and average across the constituent layers of each block $\mathcal{B}^{(k)}$ to derive the corresponding block-level base feature $f^{(k)}(x)$. Each base feature is then mapped to a high-dimensional randomized space via Eq. (1) to obtain the projected vector $\mathbf{z}^{(k)} \in \mathbb{R}^D$. Finally,

stacking these vectors across all N_t samples forms the block-specific matrix $\mathbf{Z}_t^{(k)} \in \mathbb{R}^{N_t \times D}$, thereby providing K distinct depth-aware views of $\mathcal{D}_t$.

Shallow-to-Deep Residual Learning Instead of fitting a single global classifier, ReProF sequentially estimates K block-specific prototype matrices $\{\mathbf{W}_t^{(1)}, \dots, \mathbf{W}_t^{(K)}\}$ via layered refinement: the first block fits the ground-truth labels, and each subsequent block models the prediction residual of its precursor. Ideally, for block k , the ridge regression objective over cumulative historical data $\mathbf{Z}_{\leq t}^{(k)} = [\mathbf{Z}_1^{(k)\top}, \dots, \mathbf{Z}_t^{(k)\top}]^\top$ is defined as:

$$\mathbf{W}_t^{(k)} = \arg \min_{\mathbf{W}} \left\| \mathbf{Z}_{\leq t}^{(k)} \mathbf{W} - \mathbf{R}_{\leq t}^{(k)} \right\|_F^2 + \lambda \|\mathbf{W}\|_F^2. \quad (9)$$

The global residual targets $\mathbf{R}_{\leq t}^{(k)}$ are initialized as $\mathbf{R}_{\leq t}^{(1)} := \mathbf{Y}_{\leq t}$ and recursively updated for $k \geq 2$:

$$\mathbf{R}_{\leq t}^{(k)} := \mathbf{R}_{\leq t}^{(k-1)} - \left(\mathbf{Z}_{\leq t}^{(k-1)} \mathbf{W}_t^{(k-1)} \right) \odot \mathbf{M}_{\leq t}, \quad (10)$$

where $\mathbf{Y}_{\leq t}$ is the zero-padded historical label matrix, $\mathbf{M}_{\leq t}$ a binary mask isolating invalid class assignments, and $\odot$ element-wise multiplication.

To ensure rehearsal-free optimization, Eq. (9) is solved in closed-form by recursively maintaining two sufficient statistics:

$$\mathbf{W}_t^{(k)} = \left(\mathbf{G}_t^{(k)} + \lambda \mathbf{I} \right)^{-1} \mathbf{C}_t^{(k)}, \quad (11)$$

$$\mathbf{G}_t^{(k)} = \mathbf{G}_{t-1}^{(k)} + \mathbf{Z}_t^{(k)\top} \mathbf{Z}_t^{(k)}, \quad (12)$$

$$\mathbf{C}_t^{(k)} = [\mathbf{C}_{t-1}^{(k)}, \mathbf{Z}_t^{(k)\top} \mathbf{R}_t^{(k)}], \quad (13)$$

with $\mathbf{G}_0^{(k)} = \mathbf{0}$ and $\mathbf{C}_0^{(k)} = []$. To bypass historical data access, the residual target is computed locally on the current label space $\mathcal{Y}_t$. Initialized as $\mathbf{R}_t^{(1)} := \mathbf{Y}_t$, the local residuals for $k \geq 2$ are evaluated via:

$$\mathbf{R}_t^{(k)} = \mathbf{R}_t^{(k-1)} - \left(\mathbf{Z}_t^{(k-1)} \mathbf{W}_t^{(k-1)} \right)_{\mathcal{Y}_t}, \quad (14)$$

where $(\cdot)_{\mathcal{Y}_t}$ extracts columns corresponding to $\mathcal{Y}_t$. This localized target feeds directly into $\mathbf{C}_t^{(k)}$, enabling exact incremental updates without historical sample access.

Cumulative Inference At test time, prediction scores from all K block-specific prototypes are additively aggregated to yield the final attribution:

$$s_y = \sum_{k=1}^K \left(\mathbf{z}_{\text{test}}^{(k)} \right)^\top \mathbf{w}_y^{(k)}, \quad y_{\text{pred}} = \arg \max_{y \in \mathcal{Y}_{\leq t}} s_y, \quad (15)$$

where $\mathbf{z}_{\text{test}}^{(k)}$ is the projected feature of the test case in the k -th block and $\mathbf{w}_y^{(k)}$ the y -th column of $\mathbf{W}_t^{(k)}$. This formulation aligns inference with the training-stage sequential residual fitting. The whole procedure is detailed in Algorithm 2 in Appendix A.

4 Experiments and Results

4.1 Experimental Setups

Datasets. We adopt COT-Bench (Li and Wang, 2025), a benchmark specifically tailored for continual LLM attribution. The dataset spans 8 distinct text domains, comprising human-written data and machine-generated content from 19 prominent LLMs across 12 major families (see Appendix B for full statistics). To simulate real-world LLM evolution, COT-Bench is chronologically partitioned into 10 sequential learning stages, with human texts anchoring the first stage. Under this paradigm, the system incrementally trains exclusively on sources from the current stage ($\mathcal{V}_t$), but is continuously evaluated against the cumulative label space of all previously observed models ($\mathcal{V}_{\leq t}$).

Comparison Settings. We compare ReProF with two mainstream paradigms of rehearsal-free class-incremental learning: (1) *Architecture-based methods*, which expand classifier capacity by fine-tuning lightweight, stage-specific adapters to accommodate new classes while preserving past knowledge, e.g., EASE (Zhou et al., 2024), TUNA (Wang et al., 2025), and MOS (Sun et al., 2025); (2) *Prototype-based methods*, which extract explicit or analytic prototypes from frozen backbones for matching-based attribution to resist catastrophic forgetting, e.g., SimpleCIL (Zhou et al., 2025), KLDA (Momeni et al., 2025), LoRanPAC (Peng et al., 2025), LayUP (Ahrens et al., 2023), RanPAC (McDonnell et al., 2023), and GLPD (Li and Wang, 2025). We also evaluate against (3) *non-continual baselines* designed for static attribution, including T5-Sentinel (Chen et al., 2023), LeCNN (Guo et al., 2024b), DeTeCTive (Guo et al., 2024a), and Joint Learning, an idealized upper bound where the classifier is entirely retrained at each stage using cumulative historical and current data.

Evaluation Metrics. Following prior work, we adopt final accuracy A_F and average accuracy $\bar{A}$ for evaluation. Letting A_t denote the attribution accuracy at stage t , $A_F = A_{10}$ assesses the terminal performance, while $\bar{A} = \frac{1}{10} \sum_{t=1}^{10} A_t$ assesses performance across all stages.

Implementation Details. For ReProF, we employ a frozen LLaMA-3.2-1B backbone ($d = 2048$) as the feature extractor, projecting representations into a $D = 10,000$ -dimensional space to sequentially fit $K = 2$ layer blocks, with ridge coefficients λ adaptively optimized per stage. To ensure a fair comparison, all compatible baselines share this identical backbone. We directly report published results on COT-Bench to eliminate implementation bias, and use official repositories for the remaining baselines. Implementation and hyperparameter details are deferred to Appendix C.

4.2 Main Results

Table 1 reports the final (A_F) and average ($\bar{A}$) accuracy on the COT-Bench test sets, with detailed per-LLM results deferred to Appendix F. Our key observations are summarized below.

(1) *Advantages of Multi-Layer Features.* ReProF consistently outperforms its single-layer baseline, RanPAC, across all eight domains, achieving an average improvement of +2.68% in A_F and +2.31% in $\bar{A}$. This confirms that distinct attribution cues are distributed throughout intermediate layers rather than confined to the final layer.

(2) *Prototype- vs. Architecture-based Paradigms.* Despite being training-free, competitive prototype-based methods consistently outperform parameter-intensive, architecture-based baselines. This suggests that utilizing frozen features from powerful backbones inherently resists catastrophic forgetting more effectively than optimization-based adapters.

(3) *Comparison with State-of-the-Art.* ReProF establishes a new state of the art, outperforming the strongest baseline, GLPD, by +1.08% in A_F and +1.34% in $\bar{A}$ on average. Crucially, while GLPD’s overhead scales with the incremental stages, ReProF maintains constant, lightweight cost (Fig. 6).

(4) *Approaching Non-Continual Upper Bound.* ReProF surpasses all individual non-continual baselines. Remarkably, it outpaces the idealized Joint Learning upper bound by +0.17% in $\bar{A}$, while trailing by a mere 0.96% in A_F . This gain stems from Joint Learning being restricted to final-layer features, whereas ReProF exploits richer multi-layer representations via residual fusion.

4.3 Ablation Studies

We evaluate the individual components of ReProF on COT-Bench. Since layer block partitioning and prototype fusion jointly define ReProF’s multi-block inference pipeline, directly removing either

Table 1: Main results on COT-Bench (test sets) across 8 domains. All methods use LLaMA-3.2-1B, except the non-continual baselines T5-Sentinel, LeCNN, and DeTeCTive. Best results in bold.

Method	Abstracts		Books		News		Poetry		Recipes		Reddit		Reviews		Wikipedia		Avg.	
	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$
<i>Non-continual learning baselines</i>																		
T5-Sentinel	85.12	85.06	84.59	86.41	81.05	81.55	85.43	87.14	92.92	92.71	78.98	81.14	87.07	86.99	80.37	81.63	84.49	85.33
LeCNN	79.47	85.93	85.82	87.69	81.35	84.41	85.90	87.93	89.32	91.94	78.90	83.09	87.23	86.90	77.92	79.16	83.24	85.88
DeTeCTive	81.80	88.06	86.85	87.05	82.72	85.17	85.68	86.85	92.87	94.24	81.55	83.29	87.73	85.07	80.58	82.61	84.97	86.54
Joint Learning	85.73	89.38	90.25	90.66	85.47	86.97	88.87	88.94	93.87	95.48	82.13	85.51	93.30	91.54	86.27	85.61	88.24	89.26
<i>Architecture-based continual learning</i>																		
EASE	36.67	53.22	43.88	55.16	38.43	48.02	42.47	59.37	54.50	60.36	37.12	47.32	52.03	55.84	38.17	50.65	42.91	53.74
TUNA	43.60	59.99	60.23	68.83	60.85	63.58	57.92	71.72	68.33	76.27	46.37	60.68	64.60	69.49	53.73	62.53	56.95	66.64
MOS	79.57	85.96	85.70	88.11	81.82	84.55	84.83	88.07	91.10	93.09	77.90	82.45	86.83	87.88	80.45	82.48	83.53	86.57
<i>Prototype-based continual learning</i>																		
SimpleCIL	47.52	55.49	43.07	46.81	47.27	47.85	48.68	56.26	46.92	49.95	40.25	49.21	54.70	54.22	52.20	56.53	47.58	52.04
KLDA	77.95	52.17	84.18	56.23	79.05	53.68	83.68	55.25	92.58	62.92	75.53	51.63	81.03	29.18	75.82	50.60	81.23	51.46
LoRanPAC	77.53	75.10	85.15	82.70	81.57	80.56	85.00	83.35	92.77	91.06	76.63	75.28	85.60	83.75	79.55	76.86	82.98	81.08
LayUP	80.40	86.41	85.62	88.15	82.77	85.50	87.28	89.35	93.20	94.71	76.97	82.21	88.30	89.78	81.32	82.77	84.48	87.36
RanPAC	80.72	86.39	86.40	88.00	82.27	84.98	86.47	89.03	93.05	94.62	78.22	82.35	89.43	90.49	80.37	81.11	84.62	87.12
GLPD	82.22	87.03	88.02	89.33	83.38	85.05	88.33	89.77	94.08	95.43	80.83	84.29	91.13	90.91	81.77	82.87	86.22	88.09
ReProF (ours)	84.30	88.97	88.70	91.12	85.73	88.22	88.68	90.42	94.38	95.47	81.68	84.98	90.47	90.92	84.48	85.33	87.30	89.43

Table 2: Ablation results on COT-Bench averaged across five distinct backbones as frozen feature extractors. Top: comparison to naive multi-layer fusion; Middle: impact of block partitioning; Bottom: impact of fusion mechanism.

Method	Abstracts		Books		News		Poetry		Recipes		Reddit		Reviews		Wikipedia		Avg.	
	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$
ReProF	81.95	87.84	88.46	90.56	84.67	87.43	87.00	88.55	93.31	94.76	79.91	84.37	89.73	90.13	83.81	84.86	86.11	88.56
Layer Concat	80.93	85.68	86.72	87.64	83.21	84.06	86.48	87.66	93.28	94.35	77.80	82.72	88.10	88.31	81.29	80.50	84.73	86.37
Layer Mean	80.89	86.68	86.54	87.53	83.29	86.08	85.36	88.57	93.70	93.00	78.12	83.14	89.00	88.48	81.37	83.10	83.53	87.07
RanPAC	77.17	83.23	84.42	85.89	80.76	82.25	83.60	85.71	91.95	93.39	75.47	77.16	86.42	87.00	78.12	77.90	82.24	84.31
Uniform Split	81.71	86.46	88.04	89.00	77.91	85.34	86.94	88.69	90.68	89.61	79.94	84.17	89.03	89.09	83.25	84.44	84.69	87.10
Random Split	73.86	83.39	88.09	90.16	84.31	87.09	86.83	89.27	92.05	91.29	79.84	84.08	89.64	88.22	82.86	83.97	84.68	87.18
Naive Average	81.37	87.38	88.10	90.24	84.28	87.02	86.80	88.68	93.14	94.61	79.85	84.33	89.63	90.10	83.51	84.58	85.84	88.37
Reverse Residual	80.40	86.80	87.35	89.58	83.26	86.50	86.21	88.84	92.76	94.34	79.12	83.57	89.05	89.60	82.24	83.69	85.05	87.86
Deepest Block	81.19	87.11	87.55	89.85	83.88	86.85	85.80	88.40	92.68	93.74	78.90	82.83	89.51	90.01	82.69	84.25	85.28	87.88

component would not yield a functionally comparable model. We therefore adopt substitution-based ablations, replacing the proposed designs with naive alternatives under otherwise comparable settings. To ensure architectural generality and prevent scale bias, we report results averaged across five diverse backbones covering different paradigms (encoder-only and decoder-only) and parameter scales ($\leq 0.5B$ to $8B$): RoBERTa-large, Qwen-2.5-0.5B-Instruct, LLaMA-3.2-1B, LLaMA-3-8B-Instruct, and Mistral-7B-Instruct-v0.3. The results are presented in Table 2.

Comparison to Naive Multi-Layer Fusion. To justify our block-based design, we benchmark ReProF against two naive multi-layer fusion strategies implemented within the same framework: *Layer Concat* (concatenating across all layers) and *Layer*

Mean (averaging across all layers), alongside standard *RanPAC* (last layer only) for reference. As we can see, these naive strategies yield at best marginal improvements due to their indiscriminate aggregation of multi-scale semantics. In contrast, ReProF achieves substantial gains, demonstrating that hierarchical knowledge requires deliberate structuring rather than coarse pooling.

Ablation on Block Partitioning. Having established the necessity of structured layer fusion, we validate our SVCCA-based partitioning against two heuristic alternatives while keeping the rest of the ReProF pipeline fixed ($K = 2$ blocks and shallow-to-deep residual fusion): *Uniform Split* (sequentially partitioning layers into equal-sized contiguous blocks) and *Random Split* (randomly assigning layers to blocks). The pronounced performance

Table 3: Performance of ReProF and RanPAC across five backbones on COT-Bench. Backbones include RoBERTa-large, Qwen-2.5-0.5B-Instruct, LLaMA-3.2-1B, LLaMA-3-8B-Instruct, and Mistral-7B-Instruct-v0.3.

Method	Abstracts		Books		News		Poetry		Recipes		Reddit		Reviews		Wikipedia		Avg.	
	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$
<i>RanPAC with distinct backbones</i>																		
RoBERTa	73.50	81.98	81.15	85.90	79.03	83.61	79.53	83.84	89.30	91.78	72.83	79.33	84.53	87.24	75.97	79.32	79.48	84.12
Qwen-0.5B	75.02	83.06	81.53	82.99	76.70	77.95	80.15	83.22	91.22	92.68	73.08	77.60	83.80	83.98	74.30	74.53	79.48	82.00
LLaMA-1B	80.72	86.39	86.40	88.00	82.27	84.98	86.47	89.03	93.05	94.62	78.22	82.35	89.43	90.49	80.37	81.11	84.62	87.12
LLaMA-8B	81.90	86.95	88.58	89.35	85.65	86.41	86.82	88.43	94.18	95.33	79.63	82.14	90.20	90.00	83.00	81.69	86.25	87.54
Mistral-7B	74.73	77.78	84.43	83.21	80.17	78.30	85.03	84.01	92.00	92.54	73.58	74.40	84.13	83.27	76.98	72.83	81.38	80.79
Avg.	77.17	83.23	84.42	85.89	80.76	82.25	83.60	85.71	91.95	93.39	75.47	79.16	86.42	87.00	78.12	77.90	82.24	84.31
<i>ReProF with distinct backbones</i>																		
RoBERTa	74.97	83.37	84.48	88.01	80.68	84.57	81.48	84.67	90.45	92.85	74.92	80.66	86.97	88.36	79.83	81.86	81.72	85.54
Qwen-0.5B	81.90	87.75	87.43	88.98	82.68	85.07	84.88	83.92	92.58	93.76	77.52	82.92	88.27	88.24	81.85	82.62	84.64	86.66
LLaMA-1B	84.30	88.97	88.70	91.12	85.73	88.22	88.68	90.42	94.38	95.47	81.68	84.98	90.47	90.92	84.48	85.33	87.30	89.43
LLaMA-8B	86.00	90.89	92.05	93.33	88.50	90.74	90.37	92.35	95.15	96.24	84.68	87.56	92.23	92.38	88.67	88.87	89.71	91.55
Mistral-7B	82.58	88.21	89.63	91.36	85.78	88.54	89.60	91.39	94.00	95.50	80.75	85.75	90.70	90.74	84.23	85.63	87.16	89.64
Avg.	81.95	87.84	88.46	90.56	84.67	87.43	87.00	88.55	93.31	94.76	79.91	84.37	89.73	90.13	83.81	84.86	86.11	88.56

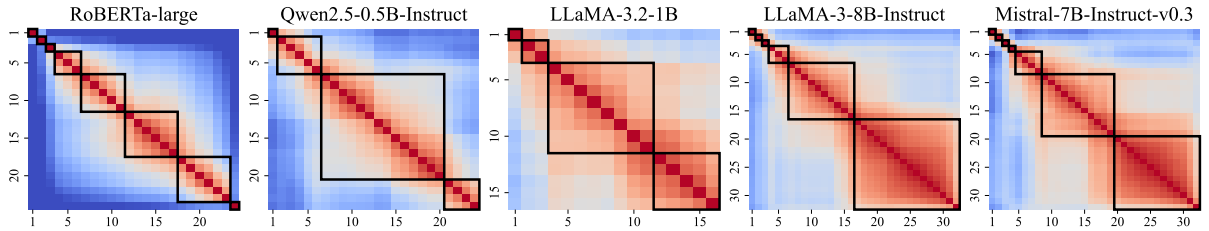


Figure 3: Layer block partitioning for five backbones on first-stage data from Abstracts. Heatmaps show pairwise SVCCA layer similarity (warmer = higher), with solid rectangles denoting identified block boundaries.

drop observed in these heuristic baselines suggests that arbitrary partitioning disrupts the natural representational coherence across layers. SVCCA consistently outperforms both counterparts, validating the critical value of aligning block boundaries with underlying semantic similarity.

Ablation on Fusion Mechanism. Finally, to verify the efficacy of our shallow-to-deep residual fusion, we evaluate three alternative mechanisms under the fixed SVCCA partitioning ($K = 2$ blocks): *Naive Average* (averaging predictions from isolated blocks), *Reverse Residual* (deep-to-shallow refinement), and *Deepest Block* (the final block alone). Table 2 shows that shallow-to-deep refinement systematically outperforms all counterparts, not only the non-fusion variant, but also the highly competitive two-block ensembles. Even within the constrained $K = 2$ setting, this targeted progressive refinement ensures consistently optimal performance.

Systematic Gains with Negligible Overhead. Although ReProF’s average gains over its closest ablated variant are modest (+0.27% in A_F and +0.19% in $\bar{A}$), paired t -tests across the eight do-

main confirm that the improvements in both metrics are statistically significant ($p < 0.05$). More importantly, the strong ablated variants share a similar partition-fusion workflow and have structural complexity comparable to ReProF. For instance, *Random Split* requires block-wise residual fusion, whereas *Naive Average* employs SVCCA-based partitioning and constructs block-specific prototype sets. Therefore, there is no significant complexity gap between ReProF and these counterparts, demonstrating that ReProF offers systematic improvements at negligible additional computational cost.

4.4 Additional Analysis

Generality across Feature Extractors. Since ReProF relies on a frozen backbone to construct prototypes, its performance naturally depends on the quality of the fixed representations. To evaluate architectural generality, we compare ReProF against its final-layer counterpart, RanPAC, using the five previously detailed backbones. Results in Table 3 show that ReProF consistently surpasses RanPAC across all the backbones, confirming that its gains

Table 4: Performance under different numbers of incremental stages on COT-Bench. Final-stage accuracy (A_F) and average-stage accuracy ($\bar{A}$) are averaged across eight domains. Best results are in bold.

	Method	$T = 1$	$T = 2$	$T = 4$	$T = 5$	$T = 10$
A_F	RanPAC	84.59	84.73	84.65	84.60	84.62
	GLPD	86.11	85.77	85.93	85.85	86.22
	ReProF	87.57	87.66	87.67	87.64	87.30
$\bar{A}$	RanPAC	84.59	85.42	87.14	85.87	87.12
	GLPD	86.11	86.21	87.80	86.74	88.09
	ReProF	87.57	88.20	89.65	88.64	89.43

are architecture-agnostic and stem from utilizing the full multi-layer structure.

Visualization of Layer Block Partitions. Fig. 3 visualizes the block partitioning results across the same five backbones evaluated previously, computed on first-stage data from the Abstracts domain with SVCCA similarity as reference. Across all backbones, the boundaries identified by our greedy algorithm largely match the representational landscapes, where intra-block layers generally exhibit higher mutual similarity and inter-block transitions are reasonably well-captured.

Robustness to Stage Configurations. To evaluate ReProF’s robustness under different continual learning settings, we vary the number of incremental stages as $T \in \{1, 2, 4, 5, 10\}$ while keeping the same 20 text sources (human-written text and 19 LLMs) and the overall label space fixed. This controlled setting changes the number of new attribution targets introduced at each stage, thereby producing different incremental granularities. Table 4 reports A_F and $\bar{A}$ averaged across the eight domains. ReProF consistently outperforms RanPAC and GLPD under every stage configuration on both metrics, demonstrating that its performance remains robust to changes in the incremental-stage configuration.

Confusion Matrix and Difficult Cases. Figure 4 presents the 20-class confusion matrix after the final incremental stage, averaged across all eight domains of COT-Bench. Two notable patterns emerge. First, most sibling models are clearly distinguishable: GPT-4/GPT-4o, MPT/MPT-Chat, and Mistral/Mistral-Chat exhibit negligible mutual confusion (at most 0.5%), whereas Cohere/Cohere-Chat is the main exception, with confusion rates of 12.8% and 11.6% in the two directions. Second,

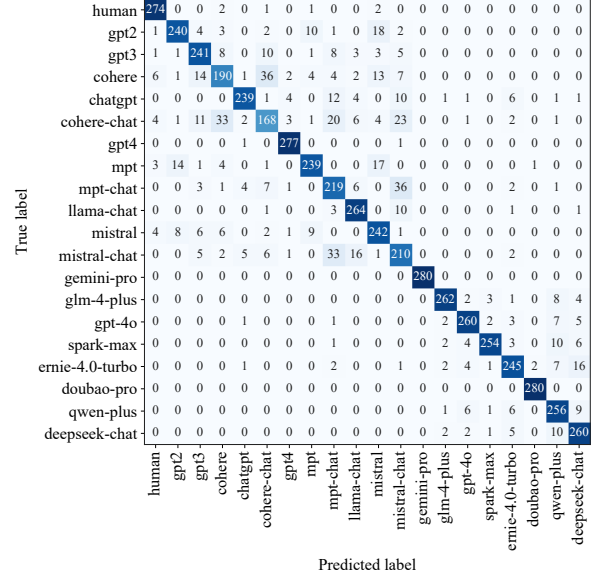


Figure 4: Average confusion matrix across all eight COT-Bench domains after the final incremental stage.

apart from this sibling pair, the largest off-diagonal errors are concentrated among cross-family chat models, particularly Cohere-Chat, MPT-Chat, and Mistral-Chat. This cross-family confusability may arise during alignment, where homogeneous instruction datasets and similar system prompts guide different models toward a standardized AI-assistant style.

Impact of Up-Projection Dimension. ReProF employs a frozen random matrix to up-project d -dimensional features into a D -dimensional space. To assess the impact of this expansion, we vary D from 3,000 to 12,000 (step size 1,000) on the Abstracts domain. Fig. 5 reveals that ReProF’s final (A_F) and average ($\bar{A}$) accuracies consistently scale with D , confirming that higher dimensionality enhances discriminative capacity. Crucially, ReProF maintains a substantial performance margin over RanPAC across the entire dimensional spectrum.

Efficiency of ReProF. We analyze computational and memory overhead of ReProF against RanPAC and GLPD on the Abstracts domain (Fig. 6). For training time, ReProF incurs marginal overhead over RanPAC during initial layer grouping; thereafter, their runtimes almost match as both are dominated by backbone extraction. Conversely, GLPD’s runtime escalates sharply to optimize classifiers for all historical-new class pairs. For storage (excluding the shared backbone), ReProF introduces a mere ~ 120.88 M extra parameters, just 9.8% of the

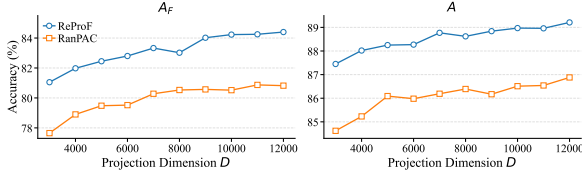


Figure 5: Final (A_F) and average ($\bar{A}$) accuracy of ReProF and RanPAC with varying D values on Abstracts. LLaMA-3.2-1B is used as the backbone.

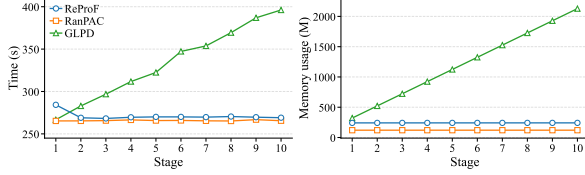


Figure 6: Computational and memory overhead of ReProF, RanPAC and GLPD on Abstracts. LLaMA-3.2-1B is used as the backbone.

LLaMA-3.2-1B size. In contrast, GLPD requires cumulative per-class statistics, causing its memory to scale linearly with the expanding class pool.

5 Related Work

LLM Attribution. The rapid advancement of LLMs has blurred the boundary between human- and machine-authored text, motivating the study of LLM attribution. This task ranges from binary human-vs-machine detection to the fine-grained identification of specific source LLMs (Li et al., 2024; Kumarage et al., 2024; Wu et al., 2025).

Early efforts primarily addressed binary detection via supervised or zero-shot methods. Supervised approaches rely on labeled data to either train classifiers using statistical features (Chen and Liu, 2023; Verma et al., 2024) or fine-tune pre-trained models for neural representations (Bhattacharjee et al., 2023, 2024). Conversely, zero-shot classifiers bypass training by extracting and thresholding specific metrics, such as entropy (Gehrmann et al., 2019), likelihood (Su et al., 2023), or probability curvature (Mitchell et al., 2023; Bao et al., 2024).

Recently, the focus has shifted toward multi-class attribution to identify the exact source LLM (Li et al., 2023; Wu et al., 2023; Chen et al., 2023). However, these methods typically assume a static set of sources, requiring costly retraining to accommodate newly emerging generators. As such, Li and Wang (2025) reformulated attribution as class-incremental learning (CIL) (Masana et al., 2022). This paradigm enables systems to incrementally learn emerging LLMs while retaining knowledge

of previous ones, perfectly aligning with the rapidly evolving LLM landscape.

Continual Learning. Continual learning (CL) enables sequential learning from data streams while mitigating catastrophic forgetting (De Lange et al., 2022). Among its primary settings (task-, domain-, and class-incremental learning (Van de Ven and Tolias, 2019)), CIL is considered the most challenging as it lacks task identifiers at inference.

Existing CIL methods broadly fall into four families (Wang et al., 2024): (1) *replay-based*, which store and reuse past samples (Aljundi et al., 2019; Chaudhry et al., 2019); (2) *regularization-based*, which constrain parameter updates or leverage distillation (Kirkpatrick et al., 2017; Li and Hoiem, 2017); (3) *architecture-based*, which allocate task-specific parameters (Zhou et al., 2024; Sun et al., 2025); (4) *prototype-based*, which leverage frozen backbones to derive order-agnostic class prototypes (Zhou et al., 2025; McDonnell et al., 2023).

Our method belongs to the prototype-based family. We improve upon final-layer-only (McDonnell et al., 2023; Li and Wang, 2025) and naive aggregation methods (Ahrens et al., 2023) by introducing a principled, sequential residual fusion across layers.

6 Conclusion

This paper introduces ReProF, a training-free continual LLM attribution framework that systematically exploits the hierarchical stylistic signatures natively embedded across a frozen backbone. Unlike prior methods that rely on final-layer features or naive fusion, thereby overlooking the distinct stylistic cues distributed throughout the model hierarchy, ReProF partitions the backbone into semantically coherent blocks and integrates them via shallow-to-deep residual refinement. Extensive experiments across 19 LLMs and 8 domains demonstrate that ReProF achieves state-of-the-art accuracy in rehearsal-free settings, consistently surpassing both existing continual methods and the non-continual joint-training upper bound. Our results confirm that effective continual attribution can be achieved by strategically harnessing the rich, hierarchical representations natively embedded within pre-trained language models.

Limitations

Despite its effectiveness, our framework presents a few avenues for further improvement. First, the current partitioning strategy for the layer blocks

(specifically the choice of K and block boundaries) is fixed at the initial stage and relies on heuristic search. Exploring more adaptive initialization during the early stage could further enhance ReProF’s automated parameter selection and structural generality. Second, as a training-free method, ReProF relies on the stylistic cues inherently present in the frozen backbone. While these representations are rich, they are constrained by the backbone’s pre-training distribution. Consequently, the framework might face challenges when the stylistic nuances of emerging LLMs deviate significantly from the feature space established during the backbone’s pre-training. Finally, ReProF assumes that the extracted features are directly separable in the projected space. In scenarios where stylistic cues are highly non-linear or inextricably entangled, more sophisticated projection head architectures might be required to further refine the discriminative capacity of the prototypes.

Acknowledgments

We would like to thank the reviewers for their insightful comments and constructive suggestions, which have significantly enhanced the quality of this work. This work is supported by the National Natural Science Foundation of China (62376033 and 62232006) and the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (JYB2025XDXM103).

References

- Kyra Ahrens, Hans Hergen Lehmann, Jae Hee Lee, and Stefan Wermter. 2023. Read between the layers: Leveraging multi-layer representations for rehearsal-free continual learning with pre-trained models. *arXiv preprint arXiv:2312.08888*.
- Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. 2019. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*.
- Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. 2024. Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *The Twelfth International Conference on Learning Representations*.
- Amrita Bhattacharjee, Tharindu Kumarage, Raha Moraffah, and Huan Liu. 2023. Conda: Contrastive domain adaptation for ai-generated text detection. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 598–610.
- Amrita Bhattacharjee, Raha Moraffah, Joshua Garland, and Huan Liu. 2024. Eagle: A domain generalization framework for ai-generated text detection. *arXiv preprint arXiv:2403.15690*.
- Arslan Chaudhry, Marc’Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. 2019. Efficient lifelong learning with a-GEM. In *International Conference on Learning Representations*.
- Yutian Chen, Hao Kang, Vivian Zhai, Liangze Li, Rita Singh, and Bhiksha Raj. 2023. Token prediction as implicit classification to identify llm-generated text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13112–13120.
- Zheng Chen and Huming Liu. 2023. Stadee: Statistics-based deep detection of machine generated text. In *International Conference on Intelligent Computing*, pages 732–743.
- Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. 2022. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44:3366–3385.
- Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. Gltr: Statistical detection and visualization of generated text. *arXiv preprint arXiv:1906.04043*.
- Xun Guo, Yongxin He, Shan Zhang, Ting Zhang, Wanquan Feng, Haibin Huang, and Chongyang Ma. 2024a. Detective: Detecting ai-generated text via multi-level contrastive learning. *Advances in Neural Information Processing Systems*, 37:88320–88347.
- Zikang Guo, Kaijie Jiao, Xingyu Yao, Yuning Wan, Haoran Li, Benfeng Xu, Licheng Zhang, Quan Wang, Yongdong Zhang, and Zhendong Mao. 2024b. Ustcbupt at semeval-2024 task 8: Enhancing machine-generated text detection via domain adversarial neural networks and llm embeddings. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 1511–1522.
- Baixiang Huang, Canyu Chen, and Kai Shu. 2025. Authorship attribution in the era of llms: Problems, methodologies, and challenges. *ACM SIGKDD Explorations Newsletter*, 26:21–43.
- Ganesh Jawahar, Benoît Sagot, and Djamel Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657.
- James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu,

- Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114:3521–3526.
- Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *International Conference on Machine Learning (ICML)*, pages 3519–3529. PMLR.
- Tharindu Kumarage, Garima Agrawal, Paras Sheth, Raha Moraffah, Aman Chadha, Joshua Garland, and Huan Liu. 2024. A survey of ai-generated text forensic systems: Detection, attribution, and characterization. *arXiv preprint arXiv:2403.01152*.
- Jooyoung Lee, Toshini Agrawal, Adaku Uchendu, Thai Le, Jinghui Chen, and Dongwon Lee. 2025. Plagbench: Exploring the duality of large language models in plagiarism generation and detection. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 7519–7534.
- Haoran Li and Quan Wang. 2025. Continual origin tracing of llm-generated text. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 479–489.
- Linyang Li, Pengyu Wang, Ke Ren, Tianxiang Sun, and Xipeng Qiu. 2023. Origin tracing and detecting of llms. *arXiv preprint arXiv:2304.14072*.
- Yafu Li, Qintong Li, Leyang Cui, Wei Bi, Zhilin Wang, Longyue Wang, Linyi Yang, Shuming Shi, and Yue Zhang. 2024. Mage: Machine-generated text detection in the wild. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 36–53.
- Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:2935–2947.
- Donald W Marquardt and Ronald D Snee. 1975. Ridge regression in practice. *The American Statistician*, 29:3–20.
- Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost Van De Weijer. 2022. Class-incremental learning: survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45:5513–5533.
- Mark D McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton Van den Hengel. 2023. Ranpac: Random projections and pre-trained models for continual learning. *Advances in Neural Information Processing Systems*, 36:12022–12053.
- Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. In *International conference on machine learning*, pages 24950–24962.
- Saleh Momeni, Sahisnu Mazumder, and Bing Liu. 2025. Continual learning using a kernel-based method over foundation models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 19528–19536.
- Ari S Morcos, Maithra Raghu, and Samy Bengio. 2018. Insights on representational similarity in neural networks with canonical correlation. *Advances in Neural Information Processing Systems (NeurIPS)*, 31.
- Liangzu Peng, Juan Elenter, Joshua Agterberg, Alejandro Ribeiro, and René Vidal. 2025. Loranpac: Low-rank random features and pre-trained models for bridging theory and practice in continual learning. In *The Thirteenth International Conference on Learning Representations*.
- Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. 2017. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. *Advances in neural information processing systems*, 30.
- Sayak Saha Roy, Poojitha Thota, Krishna Vamsi Naragam, and Shirin Nilizadeh. 2024. From chatbots to phishbots?: Phishing scam generation in commercial large language models. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 36–54.
- Rafael Alberto Rivera Soto, Kailin Koch, Aleem Khan, Barry Y Chen, Marcus Bishop, and Nicholas Andrews. 2024. Few-shot detection of machine-generated text using style representations. In *The Twelfth International Conference on Learning Representations*.
- Jinyan Su, Terry Zhuo, Di Wang, and Preslav Nakov. 2023. Detectllm: Leveraging log rank information for zero-shot detection of machine-generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12395–12412.
- Hai-Long Sun, Da-Wei Zhou, Hanbin Zhao, Le Gan, De-Chuan Zhan, and Han-Jia Ye. 2025. Mos: Model surgery for pre-trained model-based class-incremental learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 20699–20707.
- Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.
- Gido M Van de Ven and Andreas S Tolias. 2019. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*.

Vivek Verma, Eve Fleisig, Nicholas Tomlin, and Dan Klein. 2024. Ghostbuster: Detecting text ghostwritten by large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1702–1717.

Ivan Vykopal, Matúš Pikuliak, Ivan Srba, Robert Moro, Dominik Macko, and Maria Bielikova. 2024. Disinformation capabilities of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 14830–14847.

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*, 46:5362–5383.

Yan Wang, Da-Wei Zhou, and Han-Jia Ye. 2025. Integrating task-specific and universal adapters for pre-trained model-based class-incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 806–816.

Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Lidia Sam Chao, and Derek Fai Wong. 2025. A survey on llm-generated text detection: Necessity, methods, and future directions. *Computational Linguistics*, 51:275–338.

Kangxi Wu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-Seng Chua. 2023. LLMDet: A third party large language models generated text detection tool. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2113–2133.

Linglan Zhao, Xuerui Zhang, Ke Yan, Shouhong Ding, and Weiran Huang. 2024. Safe: Slow and fast parameter-efficient tuning for continual learning with pre-trained models. *Advances in Neural Information Processing Systems*, 37:113772–113796.

Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. 2025. Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. *International Journal of Computer Vision*, 133:1012–1032.

Da-Wei Zhou, Hai-Long Sun, Han-Jia Ye, and De-Chuan Zhan. 2024. Expandable subspace ensemble for pre-trained model-based class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23554–23564.

A Algorithm Details

The detailed procedures of ReProF are formalized as follows. Algorithm 1 details the greedy SVCCA-based layer block partitioning process, while Algorithm 2 details the subsequent residual prototype construction and fusion pipeline.

Algorithm 1 Layer Block Partitioning via Greedy SVCCA

Input: First-stage dataset $\mathcal{D}_1$, frozen language model with L transformer layers, similarity threshold τ

Output: A list of contiguous layer blocks ordered from deep to shallow

```

1: Extract layer representations  $\{\mathbf{H}^{(\ell)}\}_{\ell=1}^L$  on  $\mathcal{D}_1$ 
2: Initialize: blocks  $\leftarrow []$ ,  $\mathcal{B}_{\text{curr}} \leftarrow \{L\}$ ,  $\mathbf{H}_{\text{curr}} \leftarrow \mathbf{H}^{(L)}$ 
3: for  $\ell = L - 1$  to 1 do
4:    $s \leftarrow \text{SVCCA}(\mathbf{H}^{(\ell)}, \mathbf{H}_{\text{curr}})$ 
5:   if  $s \geq \tau$  then
6:      $\mathcal{B}_{\text{curr}} \leftarrow \mathcal{B}_{\text{curr}} \cup \{\ell\}$ 
7:      $\mathbf{H}_{\text{curr}} \leftarrow \frac{1}{|\mathcal{B}_{\text{curr}}|} \sum_{\ell' \in \mathcal{B}_{\text{curr}}} \mathbf{H}^{(\ell')}$ 
8:   else
9:     blocks.append( $\mathcal{B}_{\text{curr}}$ )
10:     $\mathcal{B}_{\text{curr}} \leftarrow \{\ell\}$ ,  $\mathbf{H}_{\text{curr}} \leftarrow \mathbf{H}^{(\ell)}$ 
11:   end if
12: end for
13: blocks.append( $\mathcal{B}_{\text{curr}}$ )
14: return blocks

```

Table 5: Labels for each learning stage in COT-Bench.

Learning stage	Labels
Stage 1	Human, GPT-2
Stage 2	GPT-3, Cohere
Stage 3	ChatGPT, Cohere-Chat
Stage 4	GPT-4, MPT
Stage 5	MPT-Chat, LLaMA-Chat
Stage 6	Mistral, Mistral-Chat
Stage 7	Gemini-Pro, GLM-4-Plus
Stage 8	GPT-4o, Spark-Max
Stage 9	ERNIE-4.0-Turbo, DoubaoPro
Stage 10	Qwen-Plus, DeepSeek-Chat

B Dataset Details

COT-Bench covers eight diverse text domains: Abstracts, Books, News, Poetry, Recipes, Reddit, Reviews, and Wikipedia. Each domain contains parallel data generated by 19 distinct LLMs spanning 12 major model families (e.g., GPT, LLaMA, and Gemini). To simulate real-world model evolution, these text sources are ordered chronologically by their release dates and partitioned into 10 sequential learning stages, with human-written text anchoring the first stage. The exact label space configuration for each stage is detailed in Table 5.

Data distribution within each domain is perfectly balanced, with every source contributing an identical number of samples. These samples are randomly partitioned into training, validation, and test splits. For each learning stage, the respective splits of the active LLMs are aggregated to construct the stage-level datasets. Comprehensive statistics are summarized in Table 6. The benchmark is publicly available at <https://github.com/lhr01/continual-origin-tracing>.

Algorithm 2 Residual Prototype Construction and Fusion

Input: Predefined layer blocks $\{\mathcal{B}^{(1)}, \dots, \mathcal{B}^{(K)}\}$ (shallow to deep), fixed random projection $\mathbf{R}$, nonlinearity $\phi(\cdot)$, ridge parameter $\lambda > 0$, initial sufficient statistics $\mathbf{G}_0^{(k)} = \mathbf{0}$, $\mathbf{C}_0^{(k)} = []$ for all k .

- 1: // Training (at stage t):
- 2: Receive dataset $\mathcal{D}_t$ consisting of N_t samples
- 3: Construct one-hot label matrix $\mathbf{Y}_t \in \mathbb{R}^{N_t \times |\mathcal{Y}_t|}$
- 4: **for** $k = 1$ **to** K **do**
- 5: Compute block-wise features $\mathbf{Z}_t^{(k)} \in \mathbb{R}^{N_t \times D}$ through mean-pooling, random up projection, and nonlinearity
- 6: **end for**
- 7: Initialize local residual: $\mathbf{R}_t^{(1)} \leftarrow \mathbf{Y}_t$
- 8: **for** $k = 1$ **to** K **do**
- 9: **if** $k > 1$ **then**
- 10: $\mathbf{R}_t^{(k)} \leftarrow \mathbf{R}_t^{(k-1)} - (\mathbf{Z}_t^{(k-1)} \mathbf{W}_t^{(k-1)})_{\mathcal{Y}_t}$
- 11: **end if**
- 12: $\mathbf{G}_t^{(k)} \leftarrow \mathbf{G}_{t-1}^{(k)} + \mathbf{Z}_t^{(k)\top} \mathbf{Z}_t^{(k)}$
- 13: $\mathbf{C}_t^{(k)} \leftarrow [\mathbf{C}_{t-1}^{(k)}, \mathbf{Z}_t^{(k)\top} \mathbf{R}_t^{(k)}]$
- 14: $\mathbf{W}_t^{(k)} \leftarrow (\mathbf{G}_t^{(k)} + \lambda \mathbf{I})^{-1} \mathbf{C}_t^{(k)}$
- 15: **end for**
- 16: // Inference (at test time):
- 17: Given input x_{test} , compute $\{\mathbf{z}_{\text{test}}^{(k)}\}_{k=1}^K$ as in training
- 18: **for** each class $y \in \mathcal{Y}_{\leq t}$ **do**
- 19: $s_y \leftarrow \sum_{k=1}^K (\mathbf{z}_{\text{test}}^{(k)})^\top \mathbf{w}_y^{(k)}$
- 20: **end for**
- 21: **return** $y_{\text{pred}} = \arg \max_{y \in \mathcal{Y}_{\leq t}} s_y$

Table 6: Per-LLM train/validation/test sample splits across individual domains in COT-Bench.

	Abstracts	Books	News	Poetry
Train per LLM	1,166	1,181	1,180	1,171
Valid per LLM	300	300	300	300
Test per LLM	300	300	300	300

	Recipes	Reddit	Reviews	Wikipedia
Train per LLM	1,172	1,179	643	1,179
Valid per LLM	300	300	150	300
Test per LLM	300	300	150	300

C Implementation Details

For ReProF, we adopt LLaMA-3.2-1B as a frozen feature extractor with an output dimension of $d = 2048$, and project its features into a $D = 10,000$ -dimensional space. Unless specified otherwise, we employ the SVCCA-based layer block partitioning strategy, selecting the $K = 2$ deepest blocks with an SVD rank of $r = 100$ and a grouping threshold of $\tau = 0.8$. A comprehensive comparative analysis of alternative partitioning strategies is provided in Appendix E. The ridge coefficient λ is optimized by minimizing the mean squared error on the validation split over the range $\{10^3, 10^4, \dots, 10^9\}$, independently for each stage and block.

All baseline methods share the identical LLaMA-

3.2-1B backbone, with the exception of the non-continual baselines T5-Sentinel, LeCNN, and DeTeCTive, which utilize T5-large, LLaMA-2-7B, and RoBERTa-base, respectively, as specified in their official implementations. For adapter-based methods, we maintain the identical settings as Li and Wang (2025): the adapter dimension is set to 64; the training epochs and learning rate are tuned within $\{5, 10, 15\}$ and $\{1 \times 10^{-5}, \dots, 5 \times 10^{-5}\}$, respectively; and the batch size is fixed at 16. Analytic prototype-based methods, namely RanPAC, LoRanPAC, LayUP, and GLPD, adopt the same configurations as ReProF: $d = 2048$, $D = 10,000$, and the same stage-wise λ selection procedure. All experiments are conducted on a single NVIDIA A40 48GB GPU.

D Reliability and Significance Tests

Table 1 compares ReProF with RanPAC and GLPD on COT-Bench and shows that ReProF outperforms both methods in terms of A_F and $\bar{A}$. However, all results in Table 1 are obtained using a single random seed. A single-run evaluation cannot quantify run-to-run variability and therefore leaves it unclear whether the observed performance margins remain stable across different random seeds. To assess the statistical reliability of these improvements, we repeat the experiments for all three methods using five random seeds (0, 1, 42, 727, and 3407). Table 7 reports the results averaged over the five runs, together with the corresponding standard deviations, for A_F and $\bar{A}$. As shown, ReProF consistently outperforms both baselines on both metrics, with small standard deviations demonstrating stable performance across random seeds. We further conduct paired t -tests against GLPD, which yield $p < 0.05$ for both metrics, confirming that the improvements achieved by ReProF are statistically significant.

E Analysis of Layer Block Partitioning

E.1 Similarity Metric Selection

We employ SVCCA (Raghu et al., 2017) to quantify stylistic alignment between layers in a frozen backbone. Specifically, SVCCA applies Singular Value Decomposition (SVD) to denoise network activations, subsequently utilizing Canonical Correlation Analysis (CCA) to identify shared directional components while maintaining invariance to neuron permutations. This approach enables us to systematically analyze representational similarity and derive our layer block partitioning. To justify

Table 7: Performance of ReProF, RanPAC, and GLPD across five random seeds on COT-Bench. Final-stage accuracy (A_F) and average-stage accuracy ($\bar{A}$) are reported as mean $\pm$ standard deviation. Best results are in bold.

	Method	Abstracts	Books	News	Poetry	Recipes	Reddit	Reviews	Wikipedia	Avg.
A_F	RanPAC	80.58±0.31	86.43±0.40	82.19±0.18	86.33±0.23	93.14±0.12	78.44±0.25	89.25±0.26	80.55±0.30	84.62±0.06
	GLPD	82.31±0.75	87.85±0.33	83.26±0.12	88.15±0.34	94.14±0.06	80.64±0.34	90.37±0.49	81.16±0.40	85.99±0.13
	ReProF	84.12±0.34	89.42±0.40	85.42±0.26	88.69±0.17	94.37±0.11	81.66±0.24	90.71±0.20	84.69±0.30	87.38±0.07
$\bar{A}$	RanPAC	86.54±0.21	88.20±0.20	84.73±0.22	88.92±0.20	94.60±0.04	82.58±0.15	89.66±0.49	81.00±0.17	87.03±0.06
	GLPD	87.17±0.14	89.11±0.25	85.19±0.21	89.73±0.18	95.27±0.12	84.06±0.44	90.18±0.43	81.77±0.63	87.81±0.18
	ReProF	88.95±0.08	91.18±0.09	88.15±0.18	90.52±0.07	95.44±0.05	84.91±0.19	91.04±0.15	85.39±0.15	89.45±0.03

Table 8: Comparison of SVCCA with alternative similarity metrics on COT-Bench using LLaMA-3.2-1B as the backbone. Bold indicates the best results.

Method	Abstracts		Books		News		Poetry		Recipes		Reddit		Reviews		Wikipedia		Avg.	
	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$	A_F	$\bar{A}$
Cosine	81.83	87.33	88.17	89.47	83.02	86.14	87.15	89.30	93.68	94.90	79.25	83.73	89.53	89.68	81.95	82.56	85.57	87.89
PWCCA	80.13	86.71	86.97	89.09	83.27	85.79	87.12	89.26	93.50	94.83	79.22	83.41	89.63	90.23	81.08	82.10	85.12	87.68
Linear CKA	81.37	86.62	87.38	89.23	82.72	85.81	86.40	89.26	93.22	94.72	78.23	83.13	89.63	90.23	81.68	83.02	85.08	87.75
Kernel CKA	83.50	88.72	87.15	89.26	85.63	88.15	88.37	90.23	94.50	95.46	78.22	83.11	89.50	89.82	81.82	83.06	86.09	88.48
SVCCA	84.30	88.97	88.70	91.12	85.73	88.22	88.68	90.42	94.38	95.47	81.68	84.98	90.47	90.92	84.48	85.33	87.30	89.43

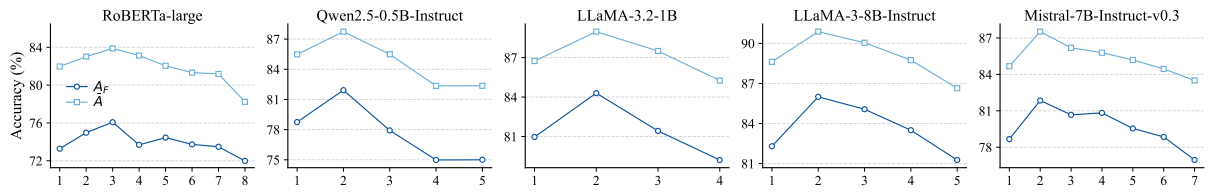


Figure 7: Performance with varying numbers of fused blocks K on Abstracts across five backbones.

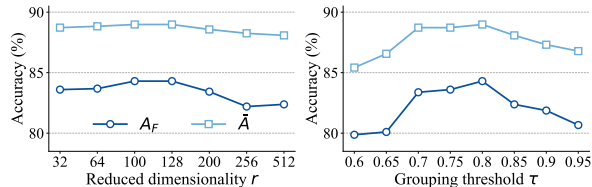


Figure 8: Performance with varying SVD rank r and grouping threshold τ on Abstracts, using LLaMA-3.2-1B as the backbone.

this choice, we benchmark SVCCA against alternative metrics, including Cosine similarity, PWCCA (Morcos et al., 2018), and CKA (Kornblith et al., 2019) (covering both linear and kernel variants).

We conduct this evaluation on the COT-Bench dataset by substituting the similarity measurement module while keeping all other components of ReProF fixed, including the hierarchical greedy partitioning procedure, the number of selected blocks, and the shallow-to-deep residual fusion. The only exception is the grouping threshold τ , which we treat as a metric-specific hyperparameter. Empirically, we observe that while SVCCA excels at $\tau =$

0.8, other metrics require a stricter threshold of $\tau = 0.95$ to effectively distinguish latent stylistic patterns. We therefore perform a grid search over $\{0.8, 0.85, 0.9, 0.95\}$ for each metric and report the best-performing result to ensure a fair comparison. Table 8 summarizes the results on COT-Bench using LLaMA-3.2-1B as the frozen backbone. We observe that SVCCA substantially and consistently outperforms alternative metrics, fully justifying its adoption as the core similarity measurement for our layer block partitioning procedure.

E.2 Sensitivity to Key Hyperparameters

Number of Fused Blocks. To evaluate the impact of the number of selected blocks K , we vary K from 1 to the maximum number of partitioned blocks. To eliminate architectural bias, we conduct this analysis across five previously evaluated backbones, including the encoder-only RoBERTa-large and four decoder-only models (i.e., Qwen2.5-0.5B-Instruct, LLaMA-3.2-1B, LLaMA-3-8B-Instruct, and Mistral-7B-Instruct-v0.3). Fig. 7 reports the results on the Abstracts domain of COT-

Table 9: Stage-wise layer block partitions on the Abstracts domain using LLaMA-3.2-1B ($L = 16$). “All stages” denotes the partition computed from pooled data across all ten stages.

Learning stage	Block 1	Block 2	Block 3	Block 4
Stage 1	$L1$	$L2-L3$	$L4-L11$	$L12-L16$
Stage 2	$L1$	$L2-L3$	$L4-L10$	$L11-L16$
Stage 3	$L1$	$L2$	$L3-L9$	$L10-L16$
Stage 4	$L1$	$L2-L3$	$L4-L11$	$L12-L16$
Stage 5	$L1$	$L2$	$L3-L9$	$L10-L16$
Stage 6	$L1$	$L2-L3$	$L4-L10$	$L11-L16$
Stage 7	$L1$	$L2-L3$	$L4-L10$	$L11-L16$
Stage 8	$L1$	$L2-L3$	$L4-L9$	$L10-L16$
Stage 9	$L1$	$L2-L3$	$L4-L9$	$L10-L16$
Stage 10	$L1$	$L2-L3$	$L4-L9$	$L10-L16$
All stages	$L1$	$L2-L3$	$L4-L10$	$L11-L16$

Bench. We observe that a relatively small K is sufficient for optimal performance: accuracy peaks at $K = 3$ for the encoder-only RoBERTa-large and at $K = 2$ for the four decoder-only models. This trend is explained by the block visualizations in Fig. 3, which show that these optimal K configurations encapsulate the vast majority of layers across all architectures. The results indicate that the few deepest blocks already capture the most distinct stylistic cues, whereas incorporating shallower layers introduces noise or redundancy that degrades performance.

SVD Rank & Grouping Threshold. We further evaluate the impact of the SVD rank r and the grouping threshold τ in the SVCCA-based partitioning. Fig. 8 reports the performance trends on the Abstracts domain of COT-Bench by varying $r \in \{32, 64, 100, 128, 200, 256, 512\}$ and $\tau \in [0.6, 0.95]$ with a step size of 0.05. We observe that the performance optimal and stabilizes around $r = 100$, whereas excessively large ranks degrade accuracy. This trend suggests that redundant dimensions retain representation noise, thereby compromising SVCCA’s ability to capture meaningful layer similarity. Similarly, the performance stays robust for $\tau \in [0.7, 0.8]$ but decreases sharply beyond this range, indicating that extreme thresholds lead to either over-splitting or over-merging of layers, which disrupts the complementarity among partitioned blocks.

E.3 Justification of Stage-1 Static Partitioning

A core design of ReProF is that the layer block partitioning (i.e., block boundaries and the number of blocks K) is determined exclusively in the initial

stage and remains strictly fixed throughout the continual pipeline. This partitioning strategy is theoretically necessitated by the rehearsal-free CIL setting. Without access to previous LLMs’ data, computing a globally optimal partition at each incremental stage is impossible. Alternatively, dynamically re-computing boundaries using solely the current stage’s data introduces a critical feature semantic inconsistency. Specifically, shifting the boundaries alters the underlying feature aggregation logic, rendering newly extracted features semantically incompatible with historical ones, thereby making it infeasible to update a unified classifier within a consistent feature space.

To assess the stability of the stage-1 partition, we recompute the SVCCA block boundaries at all ten incremental stages on Abstracts using frozen LLaMA-3.2-1B ($L = 16$). As shown in Table 9, the block boundaries remain remarkably stable across all stages. Using the stage-1 partition as the reference, the average RI, a standard clustering-similarity metric, is 0.86 (maximum 1.0), indicating strong alignment. These results demonstrate that the initial partition remains highly representative throughout continual learning, justifying a fixed-partition design that avoids dynamic re-partitioning overhead and potential forgetting.

F Detailed Attribution Results per LLM

Tables 10 and 11 present the fine-grained attribution accuracy for individual LLMs at the final incremental stage across the eight domains of CoT-Bench. For brevity, we present only the prototype-based methods that demonstrate competitive performance. The results indicate that at the end of the continual learning process, ReProF achieves the best performance across the majority of target LLMs.

Table 10: Attribution accuracy for individual LLMs at the final stage on the Abstracts, Books, News, Poetry, and Recipes domains of COT-Bench, using LLaMA-3.2-1B as the backbone.

	G_1	G_2	G_3	G_4	G_5	G_6	G_7	G_8	G_9	G_{10}	G_{11}	G_{12}	G_{13}	G_{14}	G_{15}	G_{16}	G_{17}	G_{18}	G_{19}	G_{20}
<i>Abstracts</i>																				
SimpleCIL	63.33	10.33	38.00	27.00	31.67	24.33	64.33	86.00	77.00	70.00	23.67	9.00	81.33	48.00	54.00	15.33	43.00	91.00	34.67	58.33
KLDA	99.33	78.33	79.67	68.67	76.67	52.67	93.67	73.00	78.67	84.00	61.33	57.00	98.67	79.00	77.33	65.67	81.33	98.67	76.67	78.67
LoRanPAC	98.00	76.67	86.33	70.00	69.33	48.33	97.67	69.33	82.33	87.67	65.00	46.00	100.00	80.67	80.33	61.67	75.00	99.33	73.67	83.00
LayUP	100.00	93.00	91.00	72.67	66.33	44.33	97.00	88.00	85.00	92.67	71.67	45.67	100.00	85.33	72.67	63.33	82.33	99.33	71.33	86.33
RanPAC	100.00	91.67	90.33	72.00	71.67	45.00	98.33	85.33	84.00	90.33	72.67	50.67	100.00	81.33	78.67	62.33	80.33	99.33	73.33	87.00
GLPD	100.00	94.33	88.67	74.00	76.33	49.67	96.00	84.67	83.33	90.00	74.00	59.00	99.67	79.67	82.33	66.33	78.33	99.33	81.67	87.00
ReProF	100.00	93.00	92.67	75.33	65.33	50.33	98.33	89.33	86.67	89.67	82.00	71.67	100.00	78.00	84.67	65.00	89.00	99.67	85.33	90.67
<i>Books</i>																				
SimpleCIL	11.33	3.67	65.67	10.00	11.00	16.67	72.67	88.00	75.00	76.67	11.00	43.67	44.00	91.67	41.67	18.33	15.33	89.67	31.00	44.33
KLDA	87.00	76.00	87.33	61.00	85.33	66.67	96.67	71.00	81.67	86.00	76.00	74.33	99.33	98.67	89.67	95.67	82.67	99.00	88.33	81.33
LoRanPAC	92.00	79.33	90.00	59.33	85.00	62.67	98.67	73.33	82.33	90.67	77.00	72.67	99.67	99.00	91.67	95.33	79.33	99.67	89.33	85.00
LayUP	97.00	88.00	92.33	61.00	88.00	55.00	99.00	74.33	75.67	93.00	69.00	69.00	99.67	100.00	94.00	97.00	79.33	99.67	93.33	88.00
RanPAC	96.00	83.00	90.33	62.00	83.67	64.00	99.33	76.67	81.00	91.67	78.33	72.67	99.67	99.33	94.33	95.67	82.33	99.67	91.67	86.67
GLPD	95.00	86.00	91.00	65.67	87.33	71.33	98.33	79.33	82.33	89.33	85.33	79.33	99.67	98.33	93.00	97.33	85.67	99.67	90.67	85.67
ReProF	95.67	83.00	92.33	69.67	84.33	62.33	99.67	85.33	82.67	90.00	90.33	79.67	99.67	99.00	94.33	95.67	84.33	99.67	95.67	91.33
<i>News</i>																				
SimpleCIL	15.00	8.67	39.00	10.67	57.67	18.00	83.67	86.67	68.33	86.33	11.33	31.67	55.33	47.33	32.33	45.67	48.00	85.00	51.33	63.33
KLDA	92.67	67.00	76.33	45.00	87.33	51.33	96.00	75.33	75.33	89.33	62.00	58.00	99.33	78.00	86.67	89.33	79.00	98.67	84.33	90.00
LoRanPAC	99.00	81.33	78.33	47.67	89.67	51.33	99.00	81.33	75.67	94.00	66.00	53.33	100.00	78.67	89.67	91.33	72.33	100.00	89.33	93.00
LayUP	99.33	85.67	83.00	54.00	90.67	50.67	99.33	82.00	76.00	96.33	60.67	48.67	100.00	84.00	90.67	94.33	70.00	100.00	92.00	98.00
RanPAC	99.67	81.00	78.33	51.00	89.33	53.33	99.67	83.00	75.33	95.33	69.00	56.33	100.00	78.00	88.67	91.00	71.00	100.00	91.67	93.67
GLPD	98.33	82.33	80.33	57.00	92.33	53.67	100.00	83.00	72.00	94.33	68.33	58.00	100.00	83.00	91.00	92.67	81.00	100.00	90.00	90.33
ReProF	99.67	80.33	83.67	57.67	91.33	61.33	99.33	87.00	77.33	96.33	78.67	68.33	100.00	80.00	92.67	95.00	73.67	100.00	95.00	97.33
<i>Poetry</i>																				
SimpleCIL	31.00	13.00	79.67	19.67	38.33	44.00	78.00	87.67	18.33	34.67	4.33	39.00	85.67	77.67	64.00	45.67	23.67	85.33	39.67	64.33
KLDA	91.00	77.33	85.67	66.33	85.67	66.67	98.33	65.67	61.33	87.33	64.67	64.33	98.00	98.00	98.00	94.33	90.33	98.67	90.33	91.67
LoRanPAC	95.33	83.00	87.33	72.67	86.67	65.33	98.67	74.67	57.33	91.67	66.33	67.00	98.00	98.00	99.33	93.67	81.67	99.67	92.00	93.00
LayUP	97.33	90.67	89.00	80.67	92.00	68.33	99.33	79.00	62.67	95.33	66.33	69.33	98.00	99.00	99.33	97.33	78.67	100.00	90.67	92.67
RanPAC	97.00	87.00	91.00	77.00	88.33	66.67	99.00	78.00	60.33	92.33	75.67	66.00	98.33	98.67	98.67	95.00	78.33	99.33	90.00	92.67
GLPD	98.67	89.33	93.00	75.33	91.33	69.00	99.33	80.00	65.00	92.00	77.00	67.67	98.00	98.67	99.00	96.67	89.00	100.00	94.67	93.00
ReProF	97.33	85.67	90.00	79.33	88.67	66.00	99.00	83.67	55.33	91.00	84.00	83.33	98.00	99.00	99.33	95.67	88.67	100.00	95.67	94.00
<i>Recipes</i>																				
SimpleCIL	44.67	11.67	68.33	27.33	24.00	13.67	48.67	79.33	25.00	85.33	10.67	19.00	90.00	90.67	33.33	42.67	91.00	71.00	31.00	31.00
KLDA	99.33	94.33	94.33	79.00	93.33	90.33	93.00	83.00	90.00	98.67	86.67	85.33	99.67	97.67	96.67	90.33	97.00	99.33	90.33	93.33
LoRanPAC	100.00	93.67	94.67	81.67	90.67	83.00	97.00	87.33	91.00	99.00	89.67	80.67	100.00	98.67	98.67	90.33	98.00	99.67	88.33	93.67
LayUP	99.67	95.67	97.33	83.33	89.33	87.67	98.00	88.33	89.33	99.67	85.33	80.33	100.00	98.67	98.33	91.33	98.00	99.33	88.67	95.67
RanPAC	100.00	95.67	96.00	83.00	89.67	83.33	95.67	90.67	90.33	99.00	89.00	82.00	100.00	98.33	98.33	90.67	98.00	99.33	89.00	93.00
GLPD	100.00	95.67	96.33	85.67	95.00	85.67	97.00	93.00	89.00	99.33	90.33	85.67	100.00	98.00	98.67	93.33	97.67	99.33	88.67	93.33
ReProF	100.00	95.33	96.00	77.00	90.00	88.67	96.67	93.67	91.33	99.00	95.67	87.67	100.00	97.33	98.67	95.00	98.00	99.33	91.00	97.33

Table 11: Attribution accuracy for individual LLMs at the final stage on the Reddit, Reviews, and Wikipedia domains of COT-Bench, using LLaMA-3.2-1B as the backbone.

	G_1	G_2	G_3	G_4	G_5	G_6	G_7	G_8	G_9	G_{10}	G_{11}	G_{12}	G_{13}	G_{14}	G_{15}	G_{16}	G_{17}	G_{18}	G_{19}	G_{20}
Reddit																				
SimpleCIL	60.33	24.00	35.33	14.33	19.00	15.33	33.33	79.00	81.33	35.33	12.00	23.67	55.33	69.33	24.33	37.67	29.00	74.00	47.00	35.33
KLDA	92.67	66.33	75.67	52.00	77.67	50.00	90.33	58.67	75.67	89.00	66.33	53.00	97.00	92.67	80.33	78.67	72.00	95.00	76.00	71.67
LoRanPAC	96.33	73.33	76.67	51.67	75.67	40.33	93.33	60.67	77.00	91.33	67.67	53.00	98.33	94.67	77.00	83.00	71.33	97.00	78.00	75.33
LayUP	98.67	87.00	75.33	54.67	68.00	28.00	96.00	66.67	83.00	95.33	64.33	48.00	99.33	98.00	74.00	80.67	72.33	97.00	74.33	78.67
RanPAC	98.00	77.33	77.33	54.33	76.33	39.67	95.00	65.00	80.67	92.67	73.67	53.33	98.67	97.00	75.33	81.00	76.33	97.67	78.00	77.00
GLPD	98.00	79.33	80.33	60.00	81.67	46.67	95.00	70.33	81.33	94.67	77.00	56.67	99.33	98.33	76.33	87.33	80.00	97.00	77.67	79.67
ReProF	97.33	75.33	75.00	61.67	74.33	45.00	98.33	72.00	84.33	95.67	81.00	62.00	99.00	98.00	80.00	83.67	81.00	98.67	86.00	85.33
Reviews																				
SimpleCIL	6.67	6.67	66.00	16.67	65.33	20.00	86.67	88.67	68.00	78.00	22.00	52.67	54.67	64.67	42.67	48.67	78.00	88.00	72.67	67.33
KLDA	70.67	66.67	78.67	49.33	95.33	58.67	96.67	64.67	80.67	88.00	50.67	76.67	98.00	95.33	84.67	96.00	78.67	98.67	94.00	98.67
LoRanPAC	84.67	79.33	85.33	56.00	96.00	66.00	98.67	70.67	80.00	93.33	60.67	76.67	99.33	97.33	90.67	96.67	91.33	100.00	94.00	96.67
LayUP	89.33	87.33	91.33	65.33	99.33	52.67	99.33	87.33	79.33	97.33	65.33	76.67	100.00	99.33	94.67	98.67	90.67	100.00	94.00	98.00
RanPAC	92.67	85.33	89.33	69.33	98.67	63.33	99.33	88.00	82.00	97.33	72.67	81.33	100.00	98.00	92.67	96.67	93.33	100.00	96.00	98.00
GLPD	91.33	90.00	90.00	66.00	98.00	68.00	99.33	84.00	86.67	96.67	78.00	83.33	100.00	98.67	94.67	97.33	95.33	100.00	94.67	98.67
ReProF	86.67	87.33	88.67	70.00	98.67	63.33	100.00	86.00	84.00	94.67	80.00	83.33	100.00	98.00	96.00	97.33	97.33	100.00	97.33	98.67
Wikipedia																				
SimpleCIL	64.33	17.00	55.67	13.67	71.00	12.33	74.33	78.33	23.67	82.33	32.33	12.00	76.00	64.67	67.33	66.00	56.00	91.00	36.00	50.00
KLDA	96.33	77.33	60.67	42.33	89.33	40.67	89.33	71.33	55.67	82.00	71.33	48.33	95.00	93.00	83.67	91.00	76.00	99.33	77.33	76.33
LoRanPAC	97.67	83.33	64.67	46.00	93.67	38.67	94.67	76.33	62.33	90.00	77.67	45.33	98.00	93.00	93.00	96.67	79.67	99.67	78.33	80.33
LayUP	99.67	90.67	74.67	50.67	95.67	37.33	98.00	78.00	66.67	90.00	72.33	33.33	98.33	98.33	96.67	97.67	86.67	100.00	83.33	78.33
RanPAC	98.33	85.00	67.33	48.00	95.00	40.33	95.67	75.33	62.67	91.33	78.67	43.00	98.67	98.33	92.33	97.33	82.33	100.00	78.33	79.33
GLPD	99.00	85.00	63.67	48.33	94.67	41.67	96.67	76.67	60.00	88.33	81.00	54.33	98.67	98.67	91.67	97.33	79.33	100.00	78.33	85.33
ReProF	98.33	83.67	68.67	52.67	92.67	44.33	98.00	82.67	65.33	93.67	92.33	65.67	99.00	98.00	96.67	97.33	89.33	99.67	84.33	87.33