VISUAL BRANCH IS WHAT YOU NEED FOR CLIP-BASED CLASS-INCREMENTAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

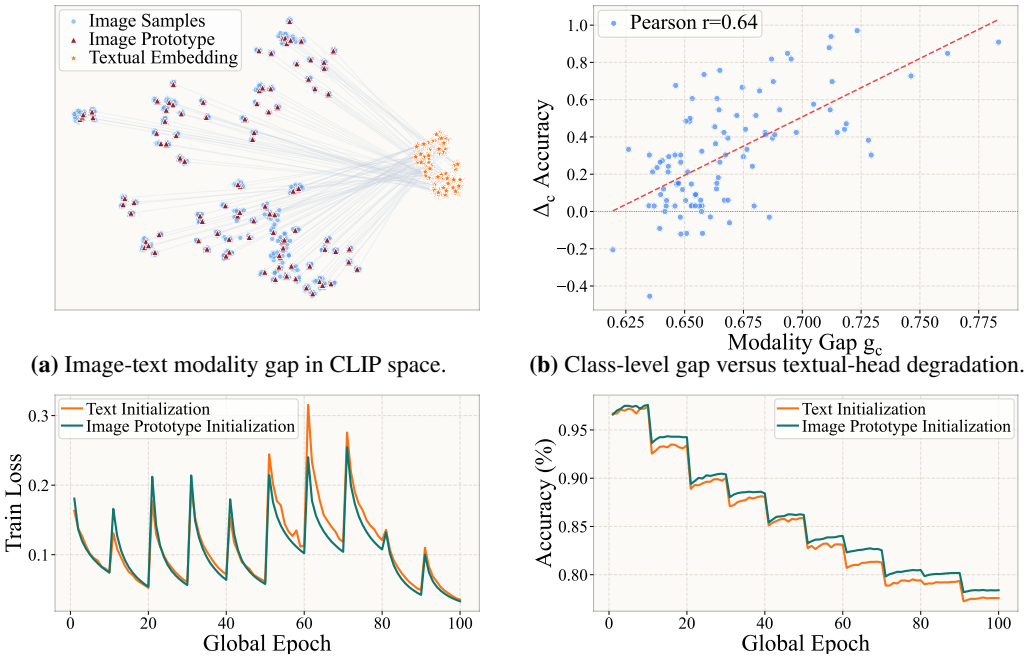
Class-Incremental Learning (CIL) requires models to recognize new classes over time without forgetting previously learned ones. With the rise of vision-language pre-training, CLIP has become a strong foundation for CIL. A common design in CLIP-based CIL is to construct textual classifier weights by encoding class-name templates with the CLIP text encoder, and then classify visual features by image-text cosine similarity. This design is appealing: since CLIP aligns images and text in a shared embedding space, textual weights appear to provide an off-the-shelf classifier for incremental classes. However, we show that this seemingly natural design is not always beneficial, as a modality gap can still separate the two modalities and make textual classifier weights deviate from visual class distributions. Empirically, under identical task-wise CIL training, initializing the cosine classifier with visual class centers yields lower loss and better incremental accuracy than using CLIP textual features. Motivated by these observations, we propose VIS (**V**isual **I**ncremental **S**VM), a visual-only method for CLIP-based CIL that removes the deployed textual branch and constructs the incremental classifier entirely in the visual space. To obtain stronger task-adaptive visual representations, VIS uses only base-session data to enhance CLIP's final visual representation with informative visual-layer features. Built on the enhanced visual representation, VIS employs a simple kernelized incremental least-squares SVM, whose classifier weights are solved in closed form from additive sufficient statistics. When new classes arrive, VIS accumulates their sufficient statistics and recomputes the classifier weights for all seen classes, enabling efficient incremental updates while preserving historical class knowledge. Extensive experiments show that VIS achieves state-of-the-art performance without a textual branch.

1 INTRODUCTION

In recent years, deep neural networks have achieved remarkable progress in visual recognition (He et al., 2016; Dosovitskiy et al., 2020). However, real-world visual systems often face dynamic environments, where new categories emerge continuously and data distributions evolve over time (Rebuffi et al., 2017; Zhao et al., 2020; Xie et al., 2026). When conventional models are updated on such non-stationary streams, they may overwrite previously acquired knowledge, leading to catastrophic forgetting (French, 1999; Serra et al., 2018; Shi et al., 2021). Class-Incremental Learning (CIL) (De Lange et al., 2021; Gao et al., 2022; Zhou et al., 2024; 2025a) addresses this challenge by requiring a model to recognize newly arriving classes while maintaining discrimination over previously learned classes. Recently, large-scale pre-trained models have become increasingly attractive for CIL by providing generalizable representations and enabling lightweight adaptation instead of training incremental models from scratch (Wang et al., 2022c; Zhou et al., 2025c). In particular, vision-language models such as CLIP (Radford et al., 2021) offer a promising foundation for CIL by combining transferable visual representations with open-vocabulary recognition capabilities (Huang et al., 2025; Zhou et al., 2025b).

Among vision-language models, CLIP is particularly appealing for CIL because it provides a powerful visual encoder together with a zero-shot classification mechanism (Radford et al., 2021). Specifically, CLIP constructs textual classifier weights by encoding class-name templates with the text encoder, and classifies visual features according to their image-text cosine similarities. This zero-shot capability naturally motivates CLIP-based CIL methods to construct incremental classifiers using

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Image-text modality gap in CLIP space.

(b) Class-level gap versus textual-head degradation.

(c) Optimization with text- and visual-based initialization.

Figure 1: Preliminary diagnosis of textual embeddings in CLIP-based CIL. (a) t-SNE visualization shows a clear separation between visual prototypes and textual embeddings in CLIP's shared image-text embedding space. (b) The class-level modality gap $g_c = 1 - \boldsymbol{\mu}_c^\top \boldsymbol{t}_c$ is positively correlated with textual-head degradation $\Delta_c = \mathrm{acc}_c^{\mathrm{NCM}} - \mathrm{acc}_c^{\mathrm{text}}$. (c) Visual-prototype initialization yields lower training loss and higher seen-class accuracy than textual initialization under the same cosine classifier.

the textual branch: some directly classify images by visual-textual similarity (Wang et al., 2023), while others use textual representations to guide prompt tuning (Lu et al., 2025), adapter learning and textual-prior guidance (Liu et al., 2023; Zhang et al., 2025), or classifier construction and representation adjustment (Huang et al., 2024; Zhou et al., 2025b). For a new class, such a strategy can derive its classifier weight directly from the class name, making the textual branch a convenient source for incremental classifier construction. However, this strategy relies on a strong assumption: textual classifier weights should provide reliable decision directions for visual samples.

To verify this assumption, we conduct a preliminary diagnosis in Figure 1 from both geometric and optimization perspectives. For each class c, we compute a normalized visual prototype $\boldsymbol{\mu}_c$ by averaging normalized CLIP visual features of its samples, and obtain the textual feature $\boldsymbol{t}_c$ by encoding its class-name template. Following the image-text modality gap in CLIP (Liang et al., 2022), we define the class-level gap as $g_c = 1 - \boldsymbol{\mu}_c^\top \boldsymbol{t}_c$, where a larger value indicates weaker visual-textual alignment. On CIFAR100 (Krizhevsky et al., 2009), Figure 1a visualizes this gap: image samples and visual prototypes cluster on the visual side, while textual features lie in a separate region of the shared CLIP space. To quantify its classification impact, we use FGVCAircraft (Maji et al., 2013) to compare two training-free classifiers: the CLIP zero-shot textual head and a nearest-class-mean classifier (Mensink et al., 2013) based on visual prototypes. We define the class-level textual-head degradation as $\Delta_c = \mathrm{acc}_c^{\mathrm{NCM}} - \mathrm{acc}_c^{\mathrm{text}}$, where larger values mean that visual prototypes outperform textual features. As shown in Figure 1b, g_c and Δ_c exhibit a positive Pearson correlation, indicating that larger modality gaps are associated with larger textual-head degradation. We further examine optimization on CIFAR100 (Krizhevsky et al., 2009) under a ten-task CIL protocol, freezing the CLIP visual encoder and training only the same cosine classifier for ten epochs per task; the only difference is whether the classifier is initialized by textual features or visual prototypes. Figure 1c reports the training loss and seen-class accuracy on the test sets of all observed classes over global epochs. Compared with visual-prototype initialization, textual initialization consistently results in higher loss and lower accuracy, indicating a less favorable optimization trajectory for the same cosine classifier. These results suggest that textual classifier weights can be geometrically misaligned with visual class distributions and harder to optimize when used to initialize the classifier. This raises a natural question: can we simply dispense with the textual branch at deployment?

Motivated by this question, we propose VIS (**V**isual **I**ncremental **S**VM), a visual-only method for CLIP-based CIL that removes the textual branch at deployment and constructs the incremental classifier entirely in the visual space. Rather than using textual embeddings as classifier weights, VIS first strengthens CLIP's final visual representation by incorporating informative visual-layer features with only base-session data. Built on the enhanced visual representation, VIS employs a simple kernelized incremental least-squares SVM, whose classifier weights are solved in closed form from additive sufficient statistics. When new classes arrive, VIS accumulates their sufficient statistics and recomputes the classifier weights for all seen classes, enabling efficient incremental updates while preserving historical class knowledge.

2 Related Work

Class-Incremental Learning. Class-incremental learning (CIL) studies how to update a model with continuously arriving new classes while maintaining its recognition ability on previously learned ones (De Lange et al., 2021; Masana et al., 2022). Early CIL methods mainly address catastrophic forgetting from several perspectives. Regularization-based approaches reduce forgetting by restricting the change of important parameters or penalizing updates in sensitive parameter directions (Kirkpatrick et al., 2017; Aljundi et al., 2018; Zenke et al., 2017). Replay-based approaches preserve information from old tasks by storing exemplar samples or synthesizing previous data distributions with generative models (Rebuffi et al., 2017; Chaudhry et al., 2018; Ostapenko et al., 2019; Xiang et al., 2019). Dynamic-architecture methods allocate additional capacity for new tasks through neuron, backbone, or prompt expansion (Yoon et al., 2017; Xu & Zhu, 2018; Wang et al., 2022a; Liu et al., 2021). Another widely used strategy is knowledge distillation, which transfers responses or representations from previous models to the current one to retain old-task knowledge (Hinton et al., 2015; Li & Hoiem, 2017; Rebuffi et al., 2017).

Pre-Trained Model-Based CIL. The emergence of large-scale pre-trained models has shifted CIL from learning representations from scratch toward adapting strong frozen backbones such as ViT (Dosovitskiy et al., 2020) and CLIP (Radford et al., 2021). This paradigm aims to exploit the generalization ability of pre-trained models while introducing only a small number of task-specific parameters (Wang et al., 2022c; Smith et al., 2023; Qi et al., 2025; Li et al., 2025). Prompt-based methods, including L2P (Wang et al., 2022c), DualPrompt (Wang et al., 2022b), and CODA-Prompt (Smith et al., 2023), adapt frozen transformers by learning or selecting prompt tokens for different tasks. Adapter-based methods insert lightweight modules into the backbone to improve task adaptation with limited parameter updates (Fukuda et al., 2025; Yu et al., 2024). For vision-language pre-trained models, CLIP-based CIL methods further leverage cross-modal alignment by using textual prototypes, prompt tuning, adapters, or language-guided representations for incremental recognition (Wang et al., 2023; Huang et al., 2024; Yu et al., 2025; Lu et al., 2025). Another line of work studies analytic or random-feature classifiers (Zhuang et al., 2022; McDonnell et al., 2023; Zhuang et al., 2023; 2024), which replace gradient-based classifier optimization with closed-form or recursive updates (Lewandowski et al., 2025; Peng et al., 2025; Momeni et al., 2025) over pre-trained features, enabling efficient exemplar-free incremental learning.

3 Preliminaries

Class-Incremental Learning. We consider the standard CIL setting with a sequence of tasks $\{\mathcal{D}_1, \ldots, \mathcal{D}_B\}$ (Rebuffi et al., 2017). Each task $\mathcal{D}_t = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N_t}$ contains samples from a class set $\mathcal{Y}_t$, where different tasks have disjoint label spaces, *i.e.*, $\mathcal{Y}_t \cap \mathcal{Y}_s = \varnothing$ for $s < t$. After learning task t, the model is evaluated over all seen classes $\mathcal{Y}_{1:t} = \bigcup_{s=1}^{t} \mathcal{Y}_s$, with cumulative class count $C_t = |\mathcal{Y}_{1:t}|$. The goal is to learn a unified classifier $f_t : \mathcal{X} \to \mathcal{Y}_{1:t}$ that recognizes both old and new classes. Following the *exemplar-free* protocol (Zhu et al., 2021; Wang et al., 2022b;c), the learner cannot store or replay samples from previous tasks. Therefore, when learning $\mathcal{D}_t$, only the current training data are accessible, while the model is required to preserve discriminative ability on $\mathcal{Y}_{1:t-1}$. We refer to the first task $\mathcal{D}_1$ as the base session and the remaining tasks as incremental sessions.

Zero-shot CLIP classification. A key capability of CLIP is its zero-shot classification mechanism: for a given set of classes, a classifier can be constructed on the fly from their names (Radford et al., 2021). Specifically, CLIP uses a visual encoder Φ_v and a textual encoder Φ_t to encode images and class-name prompts into a shared image-text embedding space. Given an image $\boldsymbol{x}_i$, the visual

encoder produces a visual embedding $\boldsymbol{t}_i = \Phi_v(\boldsymbol{x}_i) \in \mathbb{R}^d$. For each class $c \in \mathcal{Y}_{1:t}$, a class-name template, such as "a photo of a [CLASS]", is encoded by the textual encoder to produce a textual embedding $\boldsymbol{t}_c$, which serves as the classifier weight for class c. Zero-shot CLIP classifies the image by comparing the cosine similarity between the visual embedding $\boldsymbol{t}_i$ and each textual embedding $\boldsymbol{t}_c$:

$$\hat{y}(\boldsymbol{x}_i) = \arg\max_{c \in \mathcal{Y}_{1:t}} p(y = c \mid \boldsymbol{x}_i), \quad p(y = c \mid \boldsymbol{x}_i) = \frac{\exp\left(\mathrm{sim}(\boldsymbol{t}_i, \boldsymbol{t}_c)/\tau\right)}{\sum_{c' \in \mathcal{Y}_{1:t}} \exp\left(\mathrm{sim}(\boldsymbol{t}_i, \boldsymbol{t}_{c'})/\tau\right)}, \tag{1}$$

where τ is a temperature parameter.

Discussions. The zero-shot classification makes the textual branch a convenient source for constructing classifiers in CIL, but it also ties the classifier to textual embeddings. As discussed in section 1, our preliminary diagnosis shows that textual embeddings can be geometrically misaligned with visual class distributions and can hinder optimization when used to initialize the classifier. These observations suggest that textual embeddings may be unreliable anchors for the classifier. We therefore revisit the problem from a visual classification perspective: once CLIP provides a strong visual encoder, does the classifier still need to rely on the textual branch? Motivated by this question, we remove the textual branch from the classifier and construct it entirely in the visual space.

4 VIS: VISUAL INCREMENTAL SVM

To realize this classifier without the textual branch, VIS is built on two key components. First, it constructs task-adaptive visual representations by enhancing CLIP's final visual representation with multi-level visual-layer features, while keeping the CLIP visual encoder frozen. Second, it employs a kernelized incremental least-squares SVM, which computes a closed-form solution in the kernel-induced feature space from additive sufficient statistics. When new classes arrive, VIS updates these statistics and recomputes the classifier over all seen classes, enabling efficient incremental learning while preserving previously learned class information. All updates are performed entirely in the visual feature space.

4.1 TASK-ADAPTIVE VISUAL REPRESENTATION

To obtain visual representations better suited for downstream CIL, VIS does not rely solely on CLIP's final visual feature. Prior studies have shown that useful visual information is distributed across ViT/CLIP layers, and that intermediate features can complement final-layer representations for downstream recognition (Raghu et al., 2021; Ghiasi et al., 2022; Liu et al., 2025). Motivated by this observation, VIS enhances the final visual feature with multi-layer visual cues.

Visual-layer features. The frozen CLIP visual encoder consists of L transformer blocks. Given an input image $\boldsymbol{x}$, we denote the token sequence produced by the ℓ-th visual block as $\mathbf{H}^\ell(\boldsymbol{x}) = \left[\mathbf{h}^\ell_{\mathrm{cls}}(\boldsymbol{x}), \mathbf{h}^\ell_1(\boldsymbol{x}), \ldots, \mathbf{h}^\ell_N(\boldsymbol{x})\right]$. We use the CLS token of each block as its visual-layer feature:

$$\mathbf{h}_\ell(\boldsymbol{x}) = \mathbf{h}^\ell_{\mathrm{cls}}(\boldsymbol{x}) \in \mathbb{R}^{d_v}, \qquad \ell = 1, \ldots, L. \tag{2}$$

Here, $\mathbf{h}_L(\boldsymbol{x})$ is the final-layer visual feature before CLIP's cross-modal projection. For ViT-B/16, this feature is 768-dimensional. We use these raw visual features without additional normalization.

Residual fusion centered at the final feature. To adapt the visual representation to the downstream task distribution, VIS learns a residual correction to the final-layer feature from multi-layer visual features. In the full formulation, we concatenate features from all visual blocks:

$$\mathbf{m}(\boldsymbol{x}) = \mathrm{concat}\big(\mathbf{h}_1(\boldsymbol{x}), \mathbf{h}_2(\boldsymbol{x}), \ldots, \mathbf{h}_L(\boldsymbol{x})\big). \tag{3}$$

To transform the concatenated multi-layer features into a residual correction that enhances the final feature, we use a lightweight two-layer MLP as the residual mixer $\mathcal{M}$:

$$\mathcal{M}\big(\mathbf{m}(\boldsymbol{x})\big) = U\,\mathrm{GELU}(V\mathbf{m}(\boldsymbol{x}) + \mathbf{b}_V) + \mathbf{b}_U, \tag{4}$$

where $V \in \mathbb{R}^{d_h \times L d_v}$ and $U \in \mathbb{R}^{d_v \times d_h}$ are the two linear weights, and $\mathbf{b}_V, \mathbf{b}_U$ are the bias terms. The enhanced visual representation is then defined as:

$$\mathbf{u}(\boldsymbol{x}) = \mathbf{h}_L(\boldsymbol{x}) + \mathcal{M}\big(\mathbf{m}(\boldsymbol{x})\big). \tag{5}$$

We zero-initialize the final linear layer of $\mathcal{M}$, i.e., both U and $\mathbf{b}_U$, so the residual branch initially outputs zero and the fused representation starts from $\mathbf{u}(\boldsymbol{x}) = \mathbf{h}_L(\boldsymbol{x})$. During base-session training, the mixer learns only a corrective residual from other visual layers.

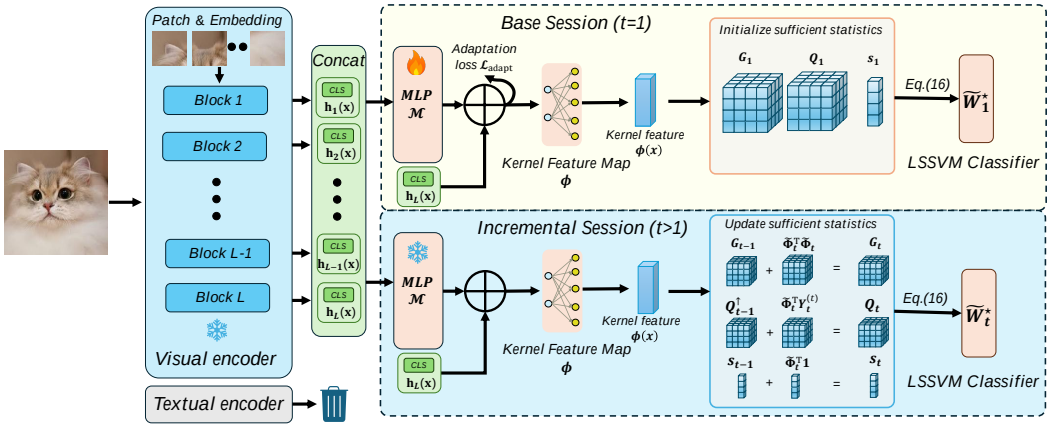


Figure 2: Illustration of VIS. The classifier is constructed entirely in the visual space: the frozen CLIP visual encoder extracts multi-layer CLS features, and the residual fusion module is trained with the adaptation loss $\mathcal{L}_{\text{adapt}}$. Residual fusion and the fixed kernel feature map then produce the kernel feature $\boldsymbol{\phi}(\boldsymbol{x})$ for classification. In the base session, VIS initializes the LS-SVM sufficient statistics $(G_1, Q_1, \mathbf{s}_1)$; in incremental sessions $(t > 1)$, it updates $(G_t, Q_t, \mathbf{s}_t)$ with new data and recomputes the classifier weights $\tilde{W}_t^\star$ in closed form.

To make this residual fusion task-adaptive, we train $\mathcal{M}$ only in the base session with an auxiliary linear classifier g_ω. Given the base-session training set $\mathcal{D}_1$, we define the adaptation loss as:

$$\mathcal{L}_{\text{adapt}} = \frac{1}{|\mathcal{D}_1|} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_1} \left[\ell_{\text{ce}}\big(g_\omega(\mathbf{u}(\boldsymbol{x})), y\big) + \lambda_{\text{adapt}} \|\mathbf{u}(\boldsymbol{x}) - \mathbf{h}_L(\boldsymbol{x})\|_2^2\right]. \tag{6}$$

Here, ℓ_{ce} denotes the cross-entropy loss, and λ_{adapt} controls the identity regularization that keeps the fused representation close to the final-layer visual feature. The CLIP visual encoder remains frozen throughout this stage. After the base session, $\mathcal{M}$ is fixed and the classifier g_ω is discarded.

4.2 INCREMENTAL KERNELIZED LS-SVM CLASSIFIER

With the enhanced visual representation, VIS learns an incremental SVM-style classifier that can incorporate new classes without being optimized only on the latest task. Several classifier choices are possible in the visual space, such as nearest-class-mean classifiers, gradient-trained linear heads, or ordinary least-squares classifiers. We choose LS-SVM because it provides discriminative one-vs-all supervision, naturally supports kernelized nonlinear decision boundaries, and admits a closed-form solution based on additive sufficient statistics. These properties make it well suited for exemplar-free CIL, where the classifier should be updated over all seen classes without storing previous samples or repeatedly fine-tuning on only the latest task.

From SVM to kernelized LS-SVM. For class c, let $y_{i,c} \in \{+1, -1\}$ be the one-vs-all target of sample $\boldsymbol{x}_i$. Given a generic feature mapping $\psi(\boldsymbol{x})$, a standard soft-margin SVM learns a separating hyperplane in the corresponding feature space:

$$\begin{aligned} \min_{\mathbf{w}_c, b_c, \{\xi_{i,c}\}} \quad & \frac{1}{2}\|\mathbf{w}_c\|_2^2 + C_{\text{svm}} \sum_{i=1}^{N} \xi_{i,c} \\ \text{s.t.} \quad & y_{i,c}\left(\mathbf{w}_c^\top \psi(\boldsymbol{x}_i) + b_c\right) \geq 1 - \xi_{i,c}, \qquad \xi_{i,c} \geq 0. \end{aligned} \tag{7}$$

Directly taking $\psi(\boldsymbol{x}) = \mathbf{u}(\boldsymbol{x})$ gives a linear classifier on the enhanced visual representation. Although $\mathbf{u}(\boldsymbol{x})$ is more task-adaptive, visually similar classes may still require nonlinear decision boundaries. We therefore introduce an explicit finite-dimensional approximation to a nonlinear kernel space: a linear classifier in the transformed space corresponds to a nonlinear decision function with respect to $\mathbf{u}(\boldsymbol{x})$. This increases classifier capacity while keeping the representation fixed for incremental updates. Instead of forming an implicit kernel matrix, we instantiate ψ with a fixed nonlinear feature map:

$$\boldsymbol{\phi}(\boldsymbol{x}) = \sigma\big(R^\top \mathbf{u}(\boldsymbol{x})\big) \in \mathbb{R}^D, \tag{8}$$

where $R \in \mathbb{R}^{d_v \times D}$ is sampled once and then fixed, and $\sigma(\cdot)$ is a nonlinear activation. This map induces a nonlinear kernel $k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^\top \boldsymbol{\phi}(\boldsymbol{x}')$, so a linear SVM on $\boldsymbol{\phi}(\boldsymbol{x})$ corresponds to an

SVM in the induced kernel space. Because ϕ is explicit and fixed, the resulting classifier remains compatible with the closed-form sufficient-statistics update introduced below. We absorb the bias term by defining:

$$\tilde{\boldsymbol{\phi}}(\boldsymbol{x}) = [\boldsymbol{\phi}(\boldsymbol{x}); 1] \in \mathbb{R}^{D+1}. \tag{9}$$

To enable efficient incremental updates, we adopt the LS-SVM relaxation (Suykens & Vandewalle, 1999), which replaces the hinge constraints with equality residuals:

$$\min_{\mathbf{w}_c, b_c, \{e_{i,c}\}} \frac{1}{2}\|\mathbf{w}_c\|_2^2 + \frac{C_{\text{svm}}}{2}\sum_{i=1}^{N} e_{i,c}^2 \qquad \text{s.t.} \quad y_{i,c}\left(\mathbf{w}_c^\top \boldsymbol{\phi}(\boldsymbol{x}_i) + b_c\right) = 1 - e_{i,c}. \tag{10}$$

Since $y_{i,c}^2 = 1$, this is equivalent to fitting classifier scores to ± 1 targets with squared residuals, while retaining the SVM-style one-vs-all coding. Stacking all seen classes together, let $\tilde{\Phi} \in \mathbb{R}^{N\times(D+1)}$ be the augmented feature matrix and $Y \in \{-1, +1\}^{N\times C}$ be the one-vs-all target matrix for C seen classes. The multi-class LS-SVM objective becomes:

$$\min_{\tilde{W}} \frac{1}{2}\text{tr}\left(\tilde{W}^\top \Gamma \tilde{W}\right) + \frac{C_{\text{svm}}}{2}\left\|\tilde{\Phi}\tilde{W} - Y\right\|_F^2, \tag{11}$$

where $\tilde{W} \in \mathbb{R}^{(D+1)\times C}$ contains classifier weights and biases, and $\Gamma = \lambda I$ is the regularizer used for all augmented feature dimensions, including the bias term. Setting the derivative to zero yields:

$$\left(\Gamma + C_{\text{svm}}\tilde{\Phi}^\top\tilde{\Phi}\right)\tilde{W} = C_{\text{svm}}\tilde{\Phi}^\top Y, \tag{12}$$

and the closed-form solution is:

$$\tilde{W}^\star = \left(\Gamma + C_{\text{svm}}\tilde{\Phi}^\top\tilde{\Phi}\right)^{-1}\left(C_{\text{svm}}\tilde{\Phi}^\top Y\right). \tag{13}$$

This form is suitable for CIL because the solution depends only on feature correlations and feature-label correlations, which can be accumulated task by task.

Additive sufficient statistics. Let $\mathcal{C}_{1:t}$ denote all classes observed up to session t, and let $C_t = |\mathcal{C}_{1:t}|$. For session s, let $\tilde{\Phi}_s$ be the augmented feature matrix of its samples. At session t, the target matrix of samples from session s is expanded to all current seen classes, denoted by $Y_s^{(t)} \in \{-1, +1\}^{N_s\times C_t}$. For each sample, the entry of its ground-truth class is $+1$, and all other entries are -1.
After session t, VIS maintains:

$$G_t = \sum_{s=1}^{t}\tilde{\Phi}_s^\top\tilde{\Phi}_s \in \mathbb{R}^{(D+1)\times(D+1)},\ Q_t = \sum_{s=1}^{t}\tilde{\Phi}_s^\top Y_s^{(t)} \in \mathbb{R}^{(D+1)\times C_t},\ \mathbf{s}_t = \sum_{s=1}^{t}\tilde{\Phi}_s^\top\mathbf{1} \in \mathbb{R}^{D+1}. \tag{14}$$

Here, G_t stores feature correlations, Q_t stores feature-label correlations, and $\mathbf{s}_t$ stores the cumulative feature sum. The statistic $\mathbf{s}_t$ is used when new classes are introduced: previous samples should serve as negative examples for these new classes, and their constant -1 contribution can be recovered from the stored feature sum without revisiting old data.

Suppose session t introduces m_t new classes. Before adding the new data contribution, we expand Q_{t-1} by appending m_t columns:

$$Q_{t-1}^{\uparrow} = \left[Q_{t-1}, -\mathbf{s}_{t-1}\mathbf{1}_{m_t}^\top\right], \tag{15}$$

where each appended column represents the negative contribution of all previous samples to a newly introduced class. We then update:

$$G_t = G_{t-1} + \tilde{\Phi}_t^\top\tilde{\Phi}_t,\ Q_t = Q_{t-1}^{\uparrow} + \tilde{\Phi}_t^\top Y_t^{(t)},\ \mathbf{s}_t = \mathbf{s}_{t-1} + \tilde{\Phi}_t^\top\mathbf{1}. \tag{16}$$

The classifier for all seen classes is recomputed as:

$$\tilde{W}_t^\star = \left(\Gamma + C_{\text{svm}}G_t\right)^{-1}\left(C_{\text{svm}}Q_t\right). \tag{17}$$

Discussions. This update is not a expansion that appends weights for new classes. Instead, it recomputes classifier weights for all seen classes from the accumulated normal equations. Thus, the classifier corresponds to fitting an LS-SVM on all seen data represented by sufficient statistics, rather than fine-tuning on only the latest task. By retaining old-class information in the accumulated statistics instead of overwriting it with new-task gradients, this mechanism mitigates catastrophic forgetting.

Table 1: Main benchmark results in terms of average accuracy $\bar{\mathcal{A}}$ and final-session accuracy $\mathcal{A}_B$. The best results are shown in bold. **All methods start from the same pre-trained CLIP backbone for fair comparison.**

Method	Aircraft				CIFAR100				Cars			
	B0 Inc10		B50 Inc10		B0 Inc10		B50 Inc10		B0 Inc10		B50 Inc10	
	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$
SimpleCIL Zhou et al. (2025a)	59.24	48.09	53.05	48.09	84.15	76.63	80.20	76.63	92.04	86.85	88.96	86.85
ACIL (Zhuang et al., 2022)	64.99	56.68	58.48	56.71	89.41	83.73	86.56	83.72	92.91	87.48	89.79	87.48
DualPrompt Wang et al. (2022b)	44.30	25.83	46.07	33.57	81.63	72.44	80.12	72.57	76.26	62.94	76.88	67.55
CODA-Prompt Smith et al. (2023)	45.98	27.69	45.14	32.28	82.43	73.43	78.69	71.58	80.21	66.47	75.06	64.19
RanPAC McDonnell et al. (2023)	69.77	60.28	63.83	60.64	89.3	83.18	86.46	83.4	93.67	89.98	91.47	89.96
RAPF Huang et al. (2024)	50.38	23.61	40.47	25.44	86.14	78.04	82.17	77.93	82.89	62.85	75.87	63.19
CLG-CBM (Yu et al., 2025)	66.05	55.93	59.25	55.39	86.58	80.15	83.59	79.28	93.25	88.76	90.11	88.19
PROOF (Zhou et al., 2025c)	63.81	56.14	59.47	57.10	86.77	79.11	83.32	79.73	90.74	86.51	88.00	85.58
BOFA (Li et al., 2026)	70.96	60.43	66.09	61.36	86.07	79.19	83.02	79.44	94.21	90.20	92.13	90.50
VIS (Ours)	**75.21**	**66.46**	**71.77**	**68.35**	**90.64**	**85.26**	**87.17**	**84.47**	**94.47**	**91.43**	**92.55**	**91.47**

Method	ImageNet-R				CUB				UCF			
	B0 Inc20		B100 Inc20		B0 Inc20		B100 Inc20		B0 Inc10		B50 Inc10	
	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$
SimpleCIL Zhou et al. (2025a)	81.06	74.48	76.84	74.48	83.81	77.52	79.75	77.52	90.44	85.68	88.12	85.68
ACIL (Zhuang et al., 2022)	82.60	77.45	79.86	77.47	84.18	78.75	80.73	78.71	98.71	97.54	98.34	97.54
DualPrompt Wang et al. (2022b)	76.21	66.65	73.22	67.58	69.89	57.46	74.40	64.84	85.21	75.82	84.31	76.35
CODA-Prompt Smith et al. (2023)	77.69	68.95	73.71	68.05	73.12	62.98	73.95	62.21	87.76	80.14	83.04	75.03
RanPAC McDonnell et al. (2023)	85.19	79.78	82.5	80.23	86.81	80.49	82.87	79.81	98.28	96.82	98.05	96.93
RAPF Huang et al. (2024)	81.26	70.48	76.10	70.23	79.09	62.77	72.82	62.93	92.28	80.33	90.31	81.55
CLG-CBM (Yu et al., 2025)	84.64	78.50	81.46	77.88	85.37	78.24	77.74	76.97	95.04	91.36	94.17	91.85
PROOF (Zhou et al., 2025c)	83.84	78.40	81.20	78.92	82.31	76.64	79.20	76.37	94.58	91.10	93.58	90.91
BOFA (Li et al., 2026)	84.53	78.77	81.60	79.12	86.66	80.58	83.18	80.79	93.19	88.71	92.60	89.43
VIS (Ours)	**86.05**	**80.58**	**82.61**	**80.38**	**87.59**	**82.02**	**84.27**	**81.98**	**99.23**	**98.37**	**99.09**	**98.75**

Method	SUN				Food				ObjectNet			
	B0 Inc30		B150 Inc30		B0 Inc10		B50 Inc10		B0 Inc20		B100 Inc20	
	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$
SimpleCIL Zhou et al. (2025a)	82.13	75.58	78.62	75.58	87.89	81.65	84.73	81.65	52.06	40.13	45.11	40.13
ACIL (Zhuang et al., 2022)	86.99	80.25	83.64	80.24	90.97	86.03	88.38	86.03	53.46	44.78	49.7	44.75
DualPrompt Wang et al. (2022b)	82.46	74.40	79.37	73.02	84.92	77.29	80.00	72.75	52.62	40.72	49.08	42.92
CODA-Prompt Smith et al. (2023)	83.34	75.71	80.38	74.17	86.18	78.78	80.98	74.13	46.49	34.13	40.57	34.13
RanPAC McDonnell et al. (2023)	87.58	81.75	84.59	81.51	90.56	85.37	87.26	84.52	57.94	45.41	53.52	48.40
RAPF Huang et al. (2024)	82.13	72.47	78.04	73.10	88.57	81.15	85.53	81.17	48.67	27.43	39.28	28.73
CLG-CBM (Yu et al., 2025)	84.85	78.09	81.58	77.79	89.12	83.05	86.76	83.85	58.53	45.11	49.80	43.56
PROOF (Zhou et al., 2025c)	83.89	77.25	80.15	76.54	90.04	84.73	87.52	84.74	56.07	43.69	48.90	43.62
BOFA (Li et al., 2026)	84.38	77.60	81.34	77.93	88.14	82.08	85.97	82.84	59.21	46.95	51.89	46.76
VIS (Ours)	**88.73**	**83.00**	**84.91**	**82.02**	**92.06**	**87.52**	**89.23**	**87.09**	**60.80**	**48.13**	**54.49**	**49.03**

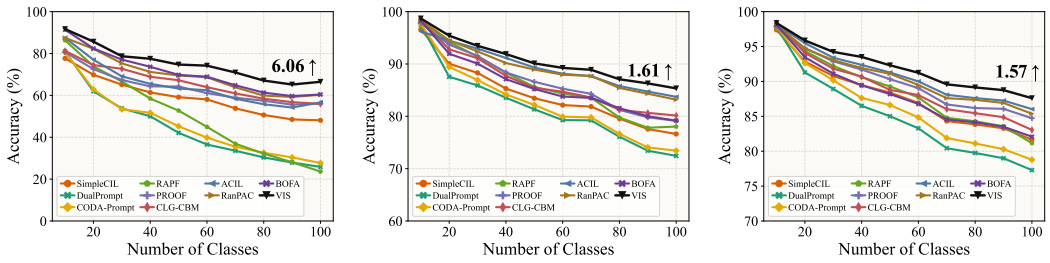
4.3 Summary of VIS

In VIS, we construct a visual-only incremental classifier by combining task-adaptive visual representations with closed-form LS-SVM updates. The residual fusion module is trained only in the base session with the adaptation loss $\mathcal{L}_{\text{adapt}}$ in Eq. 6, enhancing CLIP's final visual representation through residual fusion of multi-level visual-layer features. The classifier is built in the kernel-induced visual feature space and recomputed from additive sufficient statistics using Eq. 17.

We also determine all validation-based choices in the base session. Specifically, we split the base-session training set $\mathcal{D}_1$ into training and validation subsets. The LS-SVM regularization coefficient in Γ is selected from a predefined candidate set by minimizing the validation MSE of the one-vs-all LS-SVM targets. Although VIS is formulated with visual features from all CLIP layers, using every layer is not always necessary: it increases computation and may introduce redundant or less relevant cues. Therefore, we use the same validation split to choose a compact visual-layer subset $\mathcal{B} \subseteq \{1, \ldots, L\}$ that achieves the highest validation accuracy among candidate subsets. When a subset is used, Eq. 3 is replaced by $\mathbf{m}_{\mathcal{B}}(\boldsymbol{x}) = \text{concat}_{\ell \in \mathcal{B}} \mathbf{h}_\ell(\boldsymbol{x})$. These choices are fixed after the base session and do not use incremental-session data or any testing data.

After the base session, the residual fusion module is fixed, and each incremental session only updates the sufficient statistics $(G_t, Q_t, \mathbf{s}_t)$ and recomputes the LS-SVM weights for all seen classes. During inference, VIS relies solely on the visual branch. Given an image $\boldsymbol{x}$, we first compute the augmented feature $\tilde{\boldsymbol{\phi}}(\boldsymbol{x})$ as in Eq. 9, and use the classifier weights $\tilde{W}_t^\star$ obtained by Eq. 17. The prediction is:

$$\hat{y} = \underset{c \in \mathcal{C}_{1:t}}{\arg\max}\, \tilde{W}_t^\star[:, c]^\top \tilde{\boldsymbol{\phi}}(\boldsymbol{x}). \tag{18}$$



(a) Aircraft Base0 Inc10 (b) CIFAR100 Base0 Inc10 (c) Food Base0 Inc10

Figure 3: Incremental performance of different methods. We report the performance gap after the last incremental stage of VIS and the runner-up method at the end of the line. More results are in the supplementary.

5 EXPERIMENTS

We evaluate VIS on nine CLIP-based CIL benchmarks. We first compare it with state-of-the-art CIL methods, and then conduct ablations and additional analyses to verify the contribution of each component and the reliability of the framework. More results are provided in the appendix.

5.1 IMPLEMENTATION DETAILS

Dataset. We follow the evaluation protocol used in recent CLIP-based CIL work (Zhou et al., 2025c; 2022; Wang et al., 2022c) and report results on CIFAR100 (Krizhevsky et al., 2009), CUB200 (Wah et al., 2011), ObjectNet (Barbu et al., 2019), ImageNet-R (Hendrycks et al., 2021), FGVCAircraft (Maji et al., 2013), StanfordCars (Krause et al., 2013), Food101 (Bossard et al., 2014), SUN397 (Xiao et al., 2010), and UCF101 (Soomro et al., 2012). Following prior work (Zhou et al., 2025c), we use sampled subsets for practical partitioning: 100 classes each from CIFAR100, FGVCAircraft, StanfordCars, Food101, and UCF101; 200 classes each from CUB200, ObjectNet, and ImageNet-R; and 300 classes from SUN397.

Task splits. We use the standard 'B-m Inc-n' protocol, where m is the number of base-task classes and n is the number of classes introduced in each later task. The class order is shuffled once with seed 1993 and then shared across all methods.

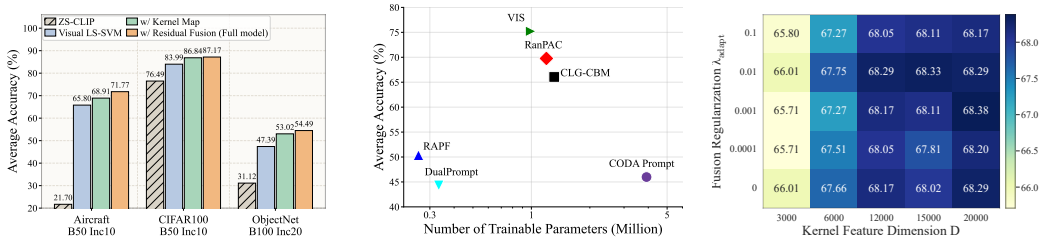
Comparison methods. We compare against representative pre-trained model-based CIL baselines, including DualPrompt (Wang et al., 2022b), CODA-Prompt (Smith et al., 2023), and SimpleCIL (Zhou et al., 2025a), as well as CLIP-based methods including RAPF (Huang et al., 2024), CLG-CBM (Yu et al., 2025), PROOF (Zhou et al., 2025c), and BOFA (Li et al., 2026). We also include ACIL (Zhuang et al., 2022) and RanPAC (McDonnell et al., 2023) as representative closed-form classifier baselines. All methods start from the same CLIP ViT-B/16 backbone.

Training details. All experiments are implemented in PyTorch (Paszke et al., 2019) and run on an NVIDIA RTX 4090 GPU. Following (Zhou et al., 2025b;c), we use the LAION-400M pre-trained CLIP ViT-B/16 model (Ilharco et al., 2021) as the visual backbone for all methods. The residual fusion module is trained only in the base session for 5 epochs using SGD with learning rate 0.01 and batch size 64. The identity regularization coefficient is set to $\lambda_{\text{adapt}} = 0.01$, and the dimension of the kernel-induced feature space is set to $D = 15000$.

Evaluation metric. Following (Rebuffi et al., 2017; Zhou et al., 2025c), we denote the model's top-1 accuracy after the b-th stage by $\mathcal{A}_b$. We report $\mathcal{A}_B$ as the final-stage accuracy and $\bar{\mathcal{A}} = \frac{1}{B}\sum_{b=1}^{B}\mathcal{A}_b$ as the mean accuracy across incremental stages.

5.2 BENCHMARK COMPARISON

We compare VIS with state-of-the-art CIL methods on benchmark datasets, and report the results in Table 1 and Figure 3. As shown, VIS consistently achieves state-of-the-art performance, demonstrating strong generalization and robustness across incremental settings. Compared with visual-only closed-form baselines such as ACIL and RanPAC, VIS shows clear gains, indicating the effectiveness of its task-adaptive visual representation and incremental LS-SVM classifier. More importantly, VIS also surpasses recent CLIP-based methods such as CLG-CBM and BOFA, which retain or exploit the textual branch for classifier construction or cross-modal guidance. These results suggest that the textual branch is not necessary for building a strong incremental classifier, and that a visual-space formulation can achieve superior performance for CLIP-based CIL.



(a) Ablation study (b) Trainable-parameter comparison (c) Parameter sensitivity

Figure 4: Ablation study, trainable-parameter comparison, and parameter sensitivity.

5.3 Further Analysis

Ablation study. We conduct component-wise ablations to analyze the contribution of each design in VIS, and report the results in Figure 4a. We start from **"ZS-CLIP"**, which uses the original CLIP zero-shot textual head. Its performance varies significantly across datasets, especially on Aircraft and ObjectNet, indicating that directly relying on textual embeddings is not always reliable for CIL. We then replace the textual head with our visual-space LS-SVM classifier on CLIP's final visual representation, denoted as **"Visual LS-SVM"**. This variant uses neither the kernel map nor residual fusion, and its large improvement shows that constructing the classifier in the visual space is the key factor. Based on this visual LS-SVM, we introduce the explicit nonlinear kernel feature map, denoted as **"w/ Kernel Map"**, which further improves accuracy by increasing classifier capacity. Finally, we add the base-session trained residual fusion module, denoted as **"w/ Residual Fusion (Full)"**. The full model achieves the best performance on all datasets, showing the benefit of enhancing CLIP's final visual representation with multi-level visual-layer features.

Parameter robustness. We analyze the sensitivity of VIS on Aircraft B0 Inc10 to two main hyperparameters: the adaptation regularization coefficient λ_{adapt} in Eq. 6 and the kernel feature dimension D in Eq. 8. We vary $\lambda_{\text{adapt}} \in \{0, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ and $D \in \{3000, 6000, 12000, 15000, 20000\}$, and report the final-stage validation accuracy in Figure 4c. As shown, VIS remains stable across a wide range of λ_{adapt} values, indicating that the residual fusion module does not rely on a carefully tuned regularization strength. Increasing D generally improves performance until saturation, showing that a sufficiently expressive kernel-induced feature space benefits the LS-SVM classifier. These results demonstrate the robustness of VIS to different hyperparameter choices.

Parameter efficiency. We compare average accuracy and trainable parameters, *i.e.*, parameters optimized by gradient descent, on Aircraft B0 Inc10 in Figure 4b. VIS achieves the best accuracy with a compact trainable parameter budget, outperforming prompt-based methods such as DualPrompt and CODA-Prompt, as well as pre-trained feature-based and CLIP-based CIL methods such as RanPAC, RAPF, and CLG-CBM. These results suggest that the gains come from the proposed visual-space formulation rather than a larger trainable parameter budget. In addition, VIS maintains non-trainable solver states $(G_t, Q_t, \mathbf{s}_t)$ for closed-form LS-SVM updates. Although these states introduce extra storage for accumulating feature statistics, they are not learnable parameters and do not require storing past images. This exemplar-free property also reduces the risk of exposing raw training data, which helps preserve data privacy in privacy-sensitive scenarios.

6 Conclusion

We revisit the role of the textual branch in CLIP-based class-incremental learning. Although CLIP enables zero-shot classification through textual embeddings, our analysis shows that these embeddings can be misaligned with visual class distributions and hinder optimization when used for classifier initialization. We therefore propose VIS, a visual-only method that removes the textual branch at deployment and constructs the incremental classifier entirely in the visual space. VIS enriches CLIP's final visual representation with multi-level visual features and employs a kernelized incremental LS-SVM whose weights are recomputed from additive sufficient statistics over all seen classes. Extensive experiments demonstrate state-of-the-art performance, showing that strong CLIP-based CIL can be achieved without relying on a deployed textual branch.

Limitations. VIS fixes the visual representation module and lifted feature space after the base session, which keeps incremental updates stable and efficient but limits further representation adaptation for later tasks. Future work may explore adaptive visual-space updates while preserving the closed-form statistic-based classifier.

AI USE STATEMENT

In this work, we used generative AI tools for editorial assistance, including checking language, notation consistency, and presentation clarity. We have not used generative AI tools for experimental execution or result generation, and the remaining required disclosure tasks are not applicable to this work. We reviewed and verified all AI-assisted work. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 139–154, 2018.

Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. *Advances in neural information processing systems*, 32, 2019.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European conference on computer vision*, pp. 446–461. Springer, 2014.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018.

Youngmin Cho and Lawrence Saul. Kernel methods for deep learning. *Advances in neural information processing systems*, 22, 2009.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

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

Takuma Fukuda, Hiroshi Kera, and Kazuhiko Kawamoto. Adapter merging with centroid prototype mapping for scalable class-incremental learning. In *Proceedings of the computer vision and pattern recognition conference*, pp. 4884–4893, 2025.

Qiankun Gao, Chen Zhao, Bernard Ghanem, and Jian Zhang. R-dfcil: Relation-guided representation learning for data-free class incremental learning. In *European Conference on Computer Vision*, pp. 423–439. Springer, 2022.

Amin Ghiasi, Hamid Kazemi, Eitan Borgnia, Steven Reich, Manli Shu, Micah Goldblum, Andrew Gordon Wilson, and Tom Goldstein. What do vision transformers learn? a visual exploration. *arXiv preprint arXiv:2212.06727*, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 8340–8349, 2021.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Linlan Huang, Xusheng Cao, Haori Lu, and Xialei Liu. Class-incremental learning with clip: Adaptive representation adjustment and parameter fusion. In *European Conference on Computer Vision*, pp. 214–231. Springer, 2024.

Linlan Huang, Xusheng Cao, Haori Lu, Yifan Meng, Fei Yang, and Xialei Liu. Mind the gap: Preserving and compensating for the modality gap in clip-based continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3777–3786, 2025.

Gabriel Ilharco, Mitchell Wortsman, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, et al. Openclip. *Zenodo*, 2021.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pp. 554–561, 2013.

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

Alex Lewandowski, Michał Bortkiewicz, Saurabh Kumar, András György, Dale Schuurmans, Mateusz Ostaszewski, and Marlos C. Machado. Learning continually by spectral regularization. In *The International Conference on Learning Representations*, 2025.

Lan Li, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Addressing imbalanced domain-incremental learning through dual-balance collaborative experts. *arXiv preprint arXiv:2507.07100*, 2025.

Lan Li, Tao Hu, Da-Wei Zhou, Jia-Qi Yang, Han-Jia Ye, and De-Chuan Zhan. Bofa: Bridge-layer orthogonal low-rank fusion for clip-based class-incremental learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 22967–22975, 2026.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Victor Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Y Zou. Mind the gap: Understanding the modality gap in multi-modal contrastive representation learning. *Advances in Neural Information Processing Systems*, 35:17612–17625, 2022.

Xialei Liu, Xusheng Cao, Haori Lu, Jia-wen Xiao, Andrew D Bagdanov, and Ming-Ming Cheng. Class incremental learning with pre-trained vision-language models. *arXiv preprint arXiv:2310.20348*, 2023.

Yajie Liu, Guodong Wang, Jinjin Zhang, Qingjie Liu, and Di Huang. Unveiling the knowledge of clip for training-free open-vocabulary semantic segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 5649–5657, 2025.

Yaoyao Liu, Bernt Schiele, and Qianru Sun. Adaptive aggregation networks for class-incremental learning. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 2544–2553, 2021.

Haodong Lu, Xinyu Zhang, Kristen Moore, Jason Xue, Lina Yao, Anton van den Hengel, and Dong Gong. Continual learning on clip via incremental prompt tuning with intrinsic textual anchors. *arXiv preprint arXiv:2505.20680*, 2025.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost Van De Weijer. Class-incremental learning: survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):5513–5533, 2022.

Mark D McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton Van den Hengel. Ranpac: Random projections and pre-trained models for continual learning. *Advances in Neural Information Processing Systems*, 36:12022–12053, 2023.

Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Distance-based image classification: Generalizing to new classes at near-zero cost. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2624–2637, 2013.

Saleh Momeni, Sahisnu Mazumder, and Bing Liu. Continual learning using a kernel-based method over foundation models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 19528–19536, 2025.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Patrick Jahnichen, and Moin Nabi. Learning to remember: A synaptic plasticity driven framework for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11321–11329, 2019.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Liangzu Peng, Juan Elenter, Joshua Agterberg, Alejandro Ribeiro, and Rene Vidal. LoRanPAC: Low-rank random features and pre-trained models for bridging theory and practice in continual learning. In *The International Conference on Learning Representations*, 2025.

Zhi-Hong Qi, Da-Wei Zhou, Yiran Yao, Han-Jia Ye, and De-Chuan Zhan. Adaptive adapter routing for long-tailed class-incremental learning. *Machine Learning*, 114(3):68, 2025.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in neural information processing systems*, 34:12116–12128, 2021.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International conference on machine learning*, pp. 4548–4557. PMLR, 2018.

Guangyuan Shi, Jiaxin Chen, Wenlong Zhang, Li-Ming Zhan, and Xiao-Ming Wu. Overcoming catastrophic forgetting in incremental few-shot learning by finding flat minima. *Advances in neural information processing systems*, 34:6747–6761, 2021.

James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11909–11919, 2023.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

Hao Sun and Da-Wei Zhou. C3box: A clip-based class-incremental learning toolbox. *arXiv preprint arXiv:2601.20852*, 2026.

Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Fu-Yun Wang, Da-Wei Zhou, Liu Liu, Han-Jia Ye, Yatao Bian, De-Chuan Zhan, and Peilin Zhao. Beef: Bi-compatible class-incremental learning via energy-based expansion and fusion. In *The eleventh international conference on learning representations*, 2022a.

Runqi Wang, Xiaoyue Duan, Guoliang Kang, Jianzhuang Liu, Shaohui Lin, Songcen Xu, Jinhu Lü, and Baochang Zhang. Attriclip: A non-incremental learner for incremental knowledge learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3654–3663, 2023.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European conference on computer vision*, pp. 631–648. Springer, 2022b.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 139–149, 2022c.

Ye Xiang, Ying Fu, Pan Ji, and Hua Huang. Incremental learning using conditional adversarial networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6619–6628, 2019.

Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pp. 3485–3492. IEEE, 2010.

Zhen-Hao Xie, Jun-Tao Tang, Yu-Cheng Shi, Han-Jia Ye, De-Chuan Zhan, and Da-Wei Zhou. Same: Stabilized mixture-of-experts for multimodal continual instruction tuning. *arXiv preprint arXiv:2602.01990*, 2026.

Ju Xu and Zhanxing Zhu. Reinforced continual learning. *Advances in neural information processing systems*, 31, 2018.

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

Jiazuo Yu, Yunzhi Zhuge, Lu Zhang, Ping Hu, Dong Wang, Huchuan Lu, and You He. Boosting continual learning of vision-language models via mixture-of-experts adapters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23219–23230, 2024.

Lu Yu, Haoyu Han, Zhe Tao, Hantao Yao, and Changsheng Xu. Language guided concept bottleneck models for interpretable continual learning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 14976–14986, 2025.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International conference on machine learning*, pp. 3987–3995. Pmlr, 2017.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11941–11952. IEEE, 2023.

Wentao Zhang, Tong Yu, Ruixuan Wang, Jianhui Xie, Emanuele Trucco, Wei-Shi Zheng, and Xiaobo Yang. Visual class incremental learning with textual priors guidance based on an adapted vision-language model. *IEEE Transactions on Multimedia*, 2025.

Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13208–13217, 2020.

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

Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. *International Journal of Computer Vision*, 133(3):1012–1032, 2025a.

Da-Wei Zhou, Kai-Wen Li, Jingyi Ning, Han-Jia Ye, Lijun Zhang, and De-Chuan Zhan. External knowledge injection for clip-based class-incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3314–3325, 2025b.

Da-Wei Zhou, Yuanhan Zhang, Yan Wang, Jingyi Ning, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Learning without forgetting for vision-language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(6):4489–4504, 2025c.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International journal of computer vision*, 130(9):2337–2348, 2022.

Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5871–5880, 2021.

Huiping Zhuang, Zhenyu Weng, Hongxin Wei, Renchunzi Xie, Kar-Ann Toh, and Zhiping Lin. Acil: Analytic class-incremental learning with absolute memorization and privacy protection. *Advances in Neural Information Processing Systems*, 35:11602–11614, 2022.

Huiping Zhuang, Zhenyu Weng, Run He, Zhiping Lin, and Ziqian Zeng. Gkeal: Gaussian kernel embedded analytic learning for few-shot class incremental task. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 7746–7755, 2023.

Huiping Zhuang, Run He, Kai Tong, Ziqian Zeng, Cen Chen, and Zhiping Lin. Ds-al: A dual-stream analytic learning for exemplar-free class-incremental learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17237–17244, 2024.

APPENDIX

In this appendix, we provide additional details and results for VIS, including the algorithmic procedure, implementation details, supplementary analyses, complete benchmark curves, and broader-impact discussion.

Section A summarizes the task-wise update of VIS, including residual fusion, fixed kernel feature mapping, sufficient-statistics updates, and the closed-form LS-SVM classifier recomputation.

Section B provides additional implementation details and further discussion on the LS-SVM classifier and kernel feature map.

Section C presents supplementary analyses of the robustness, generalization, efficiency, and forgetting behavior of VIS.

Section D provides the complete incremental learning curves under the zero-base and half-base settings.

Section E describes the compared methods used in our benchmark.

Section F studies the effect of the final classifier under fixed kernel-induced visual features, comparing NCM, Ridge, and LS-SVM.

Section G compares VIS with related classifier-based CIL methods, particularly ACIL and RanPAC, from both methodological and empirical perspectives.

Section H provides additional preliminary experiments that further support our diagnosis of textual classifier weights, including modality-gap analysis, optimization dynamics, cross-class gradient statistics, and t-SNE visualizations.

A ALGORITHMIC SUMMARY

Algorithm 1 summarizes the task-wise update of VIS. For compactness, we denote by $\mathrm{Lift}(\mathcal{D}_t; \Phi_v, \mathcal{B}, \mathcal{M}, R)$ the frozen visual pipeline that extracts selected visual-layer CLS features, applies residual fusion, computes the fixed kernel feature map, and returns the augmented feature matrix $\tilde{\Phi}_t$. The layer subset $\mathcal{B}$ and LS-SVM regularizer are selected on the base-session split as described in subsection 4.3.

B MORE IMPLEMENTATION AND REPRODUCIBILITY DETAILS

We provide implementation details that are omitted from the main text for clarity. For the base-session search of the LS-SVM regularizer λ, we use the following 16 candidate values: $\{10^{-6}, 10^{-5}, 10^{-4}, 3{\cdot}10^{-4}, 10^{-3}, 3{\cdot}10^{-3}, 10^{-2}, 3{\cdot}10^{-2}, 10^{-1}, 3{\cdot}10^{-1}, 1, 3, 10, 100, 10^{3}, 10^{4}\}$. We set $C_{\mathrm{svm}} = 10$ in all experiments; since it only changes the relative scale between the regularization and data-fitting terms, its effect can be absorbed into the selected regularization coefficient λ. For the fixed nonlinear feature map in Eq. 8, we use ReLU as $\sigma(\cdot)$, and the projection matrix R is randomly initialized once and kept fixed throughout all sessions. For the residual mixer, the hidden dimension is set to $d_h = 256$. For efficient layer subset selection, we evaluate candidate subsets using the same base-session train/validation split described in the main text. For the CLIP ViT-B/16 backbone used in our experiments, the selected subset is $\mathcal{B} = \{6, 8, 10, 12\}$, which is fixed for all incremental sessions. Both λ and $\mathcal{B}$ are selected only from the base-session training data and are never tuned on incremental-session data or test data. All datasets and pre-trained models used in this paper are publicly available research assets, and we use them only for academic evaluation while following their official terms of use and licenses.

B.1 REMARK ON THE LS-SVM CLASSIFIER

We adopt the least-squares SVM formulation (Suykens & Vandewalle, 1999) for the visual-space classifier in VIS. Although the final objective has a squared-error form, it should not be interpreted as ordinary regression on class indices or one-hot labels. Instead, each class is learned in a one-vs-all manner with ± 1 coding, where the ground-truth class is assigned $+1$ and all other seen classes

Algorithm 1 VIS task-wise update.

Input: Task data $\mathcal{D}_t$; frozen CLIP visual encoder Φ_v; selected layers $\mathcal{B}$; fusion module $\mathcal{M}$; fixed projection R; statistics $(G, Q, \mathbf{s})$; LS-SVM parameters C_{svm}, Γ.

Output: Updated statistics $(G, Q, \mathbf{s})$ and classifier $\tilde{W}_t^\star$.

1: **if** $t = 1$ **then**

2: Train $\mathcal{M}$ with $\mathcal{L}_{\text{adapt}}$ in Eq. 6, then freeze it;

3: Initialize $G \leftarrow 0,\ Q \leftarrow [\,],\ \mathbf{s} \leftarrow 0$;

4: **end if**

5: Extract augmented lifted features:

$$\tilde{\Phi}_t \leftarrow \text{Lift}(\mathcal{D}_t; \Phi_v, \mathcal{B}, \mathcal{M}, R).$$

6: Expand Q for the new classes:

$$Q \leftarrow \left[Q,\ -\mathbf{s}\mathbf{1}_{|\mathcal{Y}_t|}^\top\right].$$

7: Build $Y_t^{(t)} \in \{-1, +1\}^{N_t \times C_t}$ over $\mathcal{Y}_{1:t}$, with $+1$ for the ground-truth class and -1 otherwise;

8: Update sufficient statistics:

$$G \leftarrow G + \tilde{\Phi}_t^\top \tilde{\Phi}_t, \quad Q \leftarrow Q + \tilde{\Phi}_t^\top Y_t^{(t)}, \quad \mathbf{s} \leftarrow \mathbf{s} + \tilde{\Phi}_t^\top \mathbf{1}.$$

9: Recompute the classifier:

$$\tilde{W}_t^\star \leftarrow (\Gamma + C_{\text{svm}} G)^{-1} (C_{\text{svm}} Q).$$

10: Predict by

$$\hat{y}(\boldsymbol{x}) = \arg\max_{c \in \mathcal{Y}_{1:t}} \tilde{W}_t^\star[:, c]^\top \tilde{\boldsymbol{\phi}}(\boldsymbol{x}).$$

11: **return** $(G, Q, \mathbf{s})$ and $\tilde{W}_t^\star$.

are assigned -1. This coding preserves the SVM-style positive-versus-negative supervision, while the least-squares relaxation replaces hinge constraints with equality residuals and yields a closed-form solution. Another advantage of the SVM formulation is its *natural compatibility with kernel methods*. When an SVM is written on a feature map $\psi(\boldsymbol{x})$, its decision function can be formulated through inner products in the corresponding feature space, which induces a kernelized classifier. In our case, the explicit map $\boldsymbol{\phi}(\boldsymbol{x})$ induces the finite-dimensional kernel $k_D(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^\top \boldsymbol{\phi}(\boldsymbol{x}')$, so applying LS-SVM on $\boldsymbol{\phi}(\boldsymbol{x})$ corresponds to applying LS-SVM in the induced kernel feature space. This motivates our use of the fixed nonlinear feature map before the LS-SVM classifier.

This property is particularly useful for CIL: the closed-form system depends on the data only through feature correlations $\tilde{\Phi}^\top \tilde{\Phi}$ and feature-label correlations $\tilde{\Phi}^\top Y$, which can be accumulated across sessions as additive sufficient statistics rather than optimized by repeatedly fine-tuning on the latest task. When new classes arrive, VIS expands the one-vs-all target matrix to include the new classes and recomputes the classifier weights for all seen classes from the accumulated statistics. Therefore, the classifier corresponds to fitting an LS-SVM over all seen data represented by sufficient statistics, without storing or revisiting previous samples.

B.2 KERNEL INTERPRETATION OF THE FIXED NONLINEAR FEATURE MAP

We provide a detailed explanation of why the fixed nonlinear feature map in Eq. 8 can be interpreted from a kernel perspective. Recall that VIS maps the enhanced visual representation $\mathbf{u}(\boldsymbol{x})$ to

$$\boldsymbol{\phi}(\boldsymbol{x}) = \sigma(R^\top \mathbf{u}(\boldsymbol{x})) \in \mathbb{R}^D, \tag{19}$$

where R is sampled once and then fixed, and $\sigma(\cdot)$ is a nonlinear activation.

Finite-dimensional kernel induced by the explicit map. For any fixed R, the explicit feature map $\boldsymbol{\phi}$ defines the following finite-dimensional kernel:

$$k_D(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^\top \boldsymbol{\phi}(\boldsymbol{x}'). \tag{20}$$

This is a valid positive semi-definite kernel. To see this, consider any finite set of samples $\{\boldsymbol{x}_i\}_{i=1}^n$ and form the feature matrix

$$\Phi_D = \begin{bmatrix} \boldsymbol{\phi}(\boldsymbol{x}_1)^\top \\ \boldsymbol{\phi}(\boldsymbol{x}_2)^\top \\ \vdots \\ \boldsymbol{\phi}(\boldsymbol{x}_n)^\top \end{bmatrix} \in \mathbb{R}^{n \times D}. \tag{21}$$

The Gram matrix induced by k_D is

$$K_D = \left[k_D(\boldsymbol{x}_i, \boldsymbol{x}_j)\right]_{i,j=1}^n = \Phi_D \Phi_D^\top. \tag{22}$$

Therefore, for any coefficient vector $\boldsymbol{\alpha} \in \mathbb{R}^n$,

$$\begin{aligned} \boldsymbol{\alpha}^\top K_D \boldsymbol{\alpha} &= \boldsymbol{\alpha}^\top \Phi_D \Phi_D^\top \boldsymbol{\alpha} \\ &= \left\| \Phi_D^\top \boldsymbol{\alpha} \right\|_2^2 = \left\| \sum_{i=1}^n \alpha_i \boldsymbol{\phi}(\boldsymbol{x}_i) \right\|_2^2 \geq 0. \end{aligned} \tag{23}$$

Thus, K_D is positive semi-definite for any finite input set, and k_D is a valid kernel. Consequently, training a linear LS-SVM on $\boldsymbol{\phi}(\boldsymbol{x})$ is equivalent to training an LS-SVM in the finite-dimensional feature space induced by k_D. This argument is exact for any fixed D and does not rely on asymptotic approximation.

Connection to random-feature kernel approximation. The randomness of R further provides a connection to random-feature kernel approximation (Rahimi & Recht, 2007). Let the columns of R be independently sampled as $\mathbf{r}_1, \ldots, \mathbf{r}_D$. For analysis, consider the normalized kernel

$$\begin{aligned} \bar{k}_D(\boldsymbol{x}, \boldsymbol{x}') &= \frac{1}{D} \boldsymbol{\phi}(\boldsymbol{x})^\top \boldsymbol{\phi}(\boldsymbol{x}') \\ &= \frac{1}{D} \sum_{j=1}^D \sigma(\mathbf{r}_j^\top \mathbf{u}(\boldsymbol{x})) \sigma(\mathbf{r}_j^\top \mathbf{u}(\boldsymbol{x}')). \end{aligned} \tag{24}$$

For fixed $\boldsymbol{x}$ and $\boldsymbol{x}'$, by the law of large numbers,

$$\bar{k}_D(\boldsymbol{x}, \boldsymbol{x}') \xrightarrow[D \to \infty]{} k_\infty(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{\mathbf{r}} \left[\sigma(\mathbf{r}^\top \mathbf{u}(\boldsymbol{x})) \sigma(\mathbf{r}^\top \mathbf{u}(\boldsymbol{x}')) \right]. \tag{25}$$

Therefore, the fixed finite-dimensional feature map used by VIS can also be viewed as a Monte Carlo approximation to the nonlinear kernel k_∞. The unnormalized kernel k_D in Eq. 20 differs from $\bar{k}_D$ only by the constant factor D, which can be absorbed into the LS-SVM regularization coefficient.

ReLU activation and the arc-cosine kernel. In our implementation, σ is the ReLU activation, and R is randomly initialized once and then fixed. The finite-dimensional kernel interpretation above holds for any fixed realization of R. To connect this construction to a standard closed-form kernel, we consider the common Gaussian case, where each random direction satisfies $\mathbf{r} \sim \mathcal{N}(0, I)$. In this case, the limiting kernel in Eq. 25 corresponds to the first-order arc-cosine kernel (Cho & Saul, 2009).

Let θ be the angle between $\mathbf{u}(\boldsymbol{x})$ and $\mathbf{u}(\boldsymbol{x}')$, *i.e.*,

$$\cos\theta = \frac{\mathbf{u}(\boldsymbol{x})^\top \mathbf{u}(\boldsymbol{x}')}{\|\mathbf{u}(\boldsymbol{x})\|_2 \|\mathbf{u}(\boldsymbol{x}')\|_2}. \tag{26}$$

For ReLU activation and Gaussian random directions, we have

$$\begin{aligned} &\mathbb{E}_{\mathbf{r} \sim \mathcal{N}(0,I)} \left[\mathrm{ReLU}(\mathbf{r}^\top \mathbf{u}(\boldsymbol{x})) \mathrm{ReLU}(\mathbf{r}^\top \mathbf{u}(\boldsymbol{x}')) \right] \\ &\quad = \frac{\|\mathbf{u}(\boldsymbol{x})\|_2 \|\mathbf{u}(\boldsymbol{x}')\|_2}{2\pi} \left[\sin\theta + (\pi - \theta)\cos\theta \right]. \end{aligned} \tag{27}$$

This expression is the ReLU form of the first-order arc-cosine kernel. Thus, with Gaussian random directions and ReLU activation, the fixed nonlinear feature map provides a finite-dimensional approximation to a ReLU-induced nonlinear kernel space. For other random initializations, the exact

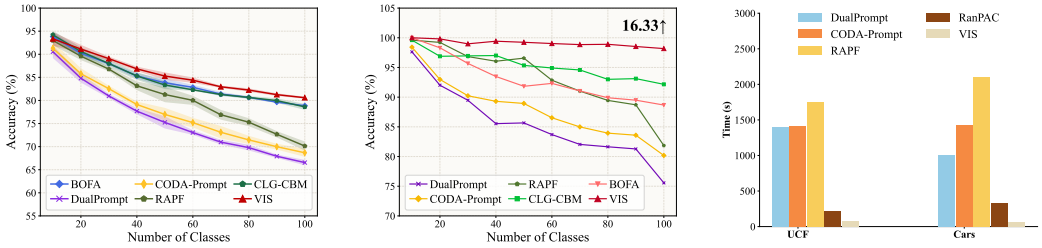
limiting kernel may differ, but any fixed realization of R still defines a valid finite-dimensional kernel through the inner product $\phi(\boldsymbol{x})^\top \phi(\boldsymbol{x}')$. Since R is fixed and never optimized, this kernelized feature space introduces no additional trainable parameters during incremental learning.

Why introduce a kernel-induced feature space. A linear classifier on $\mathbf{u}(\boldsymbol{x})$ can only form hyperplane decision boundaries in the enhanced visual space. By contrast, applying a nonlinear feature map $\phi(\boldsymbol{x}) = \sigma(R^\top \mathbf{u}(\boldsymbol{x}))$ and then learning a linear classifier on $\phi(\boldsymbol{x})$ yields decision boundaries that are nonlinear with respect to $\mathbf{u}(\boldsymbol{x})$. Specifically, the class boundary between c and c' is

$$(\mathbf{w}_c - \mathbf{w}_{c'})^\top \phi(\boldsymbol{x}) + (b_c - b_{c'}) = 0, \tag{28}$$

whose preimage in the original visual space is generally nonlinear because ϕ is nonlinear. Therefore, the kernel-induced feature space increases classifier capacity while preserving the closed-form LS-SVM solution in the explicit feature space.

Why use the explicit feature map. Although one could use the implicit kernel k_∞, doing so would require maintaining or recomputing sample-level kernel matrices over the data observed so far, which is inconvenient for CIL. Instead, VIS uses the explicit finite-dimensional feature map $\phi(\boldsymbol{x})$. This keeps the LS-SVM in a primal closed-form formulation, where the required feature correlations and feature-label correlations can be accumulated as additive sufficient statistics. Therefore, the fixed nonlinear feature map provides nonlinear classifier capacity while preserving the efficient incremental update rule of VIS without storing previous samples.



(a) Results on ImageNet-R B0 Inc20 averaged over five class-order seeds. VIS consistently outperforms the compared methods across incremental stages.

(b) Experiments when using OpenAI weights on UCF B0 Inc10. VIS consistently outperforms other methods across different backbone weights.

(c) Training time comparison on UCF B0 Inc10 and Cars B0 Inc10.

Figure 5: Additional analysis of VIS. Left: robustness to different class-order seeds. Middle: performance under different CLIP pre-trained weights. Right: training time comparison with representative baselines.

C SUPPLEMENTARY RESULTS AND ANALYSES

C.1 EVALUATION ACROSS MULTIPLE RANDOM SEEDS

The main experiments follow the standard CIL protocol (Rebuffi et al., 2017) and use the class-order seed 1993. To assess robustness to class ordering, we further run five independent splits with seeds $\{1993, 1994, 1995, 1996, 1997\}$ and report the mean accuracy with standard deviation. As shown in Figure 5a, VIS maintains the best performance on ImageNet-R B0 Inc20 across repeated runs, indicating that its advantage is stable under different class orders.

C.2 RANDOM-FEATURE ROBUSTNESS

The projection matrix R in Eq. 8 is randomly initialized once and then fixed throughout all incremental sessions. To examine whether VIS depends on a particular realization of the random feature map, we vary only the initialization seed of R, while keeping the class order and all other experimental settings unchanged. For reference, we also report representative analytic and CLIP-based baselines under the same incremental protocols. The results of VIS are reported as mean $\pm$ standard

Table 2: Random-feature robustness under different initialization seeds of R. Results of VIS are reported as mean $\pm$ standard deviation across random-feature seeds, while the other methods are included as references under the same incremental protocols.

Method	Aircraft B0 Inc10		CIFAR100 B0 Inc10	
	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$
ACIL (Zhuang et al., 2022)	64.99	56.68	89.41	83.73
RanPAC (McDonnell et al., 2023)	69.77	60.28	89.30	83.18
BOFA (Li et al., 2026)	70.96	60.43	86.07	79.19
ENGINE (Zhou et al., 2025b)	69.74	58.51	86.89	79.29
VIS (Ours)	$\mathbf{75.236 \pm 0.106}$	$\mathbf{66.520 \pm 0.277}$	$\mathbf{90.598 \pm 0.073}$	$\mathbf{85.428 \pm 0.167}$

Table 3: Generalization across different pre-trained backbones under the B0 Inc10 protocol. All methods use the same backbone within each group. We report average accuracy $\bar{\mathcal{A}}$ and final-session accuracy $\mathcal{A}_B$.

Backbone	Method	Aircraft		CIFAR100	
		$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$
DINOv2 ViT-B/14	SimpleCIL (Zhou et al., 2025a)	46.00	37.23	92.36	88.10
	RanPAC (McDonnell et al., 2023)	87.39	79.81	90.09	83.58
	VIS (Ours)	**87.68**	**80.53**	**95.17**	**91.98**
SigLIP ViT-B/16	SimpleCIL (Zhou et al., 2025a)	78.18	69.76	81.16	74.34
	RanPAC (McDonnell et al., 2023)	76.35	66.31	86.87	79.85
	VIS (Ours)	**82.68**	**75.40**	**89.36**	**83.83**

deviation across different random-feature seeds, whereas the baseline results correspond to their standard evaluations.

As shown in Table 2, VIS exhibits consistently small variations across different random feature mappings on both datasets. The standard deviations are at most 0.106 for average accuracy and 0.277 for final-session accuracy on Aircraft, and further decrease to 0.073 and 0.167 on CIFAR100. Moreover, these variations are substantially smaller than the performance margins over the compared baselines. These results indicate that the performance of VIS is stable across different samplings of R and that its gains do not depend on a favorable random feature initialization.

C.3 Parameter Robustness

For the parameter sensitivity analysis reported in the main paper, we use a validation split from the original training data. Specifically, the training data are split into training and validation subsets with a ratio of $4{:}1$, and all sensitivity results are reported as final-stage accuracies on the validation subset. The results show stable performance trends across different hyperparameter settings, further supporting the robustness of VIS.

C.4 Results with Different Backbones

Our main experiments use CLIP ViT-B/16 with LAION-400M pre-trained weights (Ilharco et al., 2021). To evaluate whether the proposed visual-only analytic framework generalizes across CLIP initializations, we further test VIS with OpenAI CLIP weights (Radford et al., 2021) on UCF B0 Inc10, as shown in Figure 5b. VIS consistently outperforms the compared methods under this alternative backbone, indicating that its advantage is not tied to a specific CLIP pre-training source.

C.5 Backbone Generalization

To examine whether VIS is tied to the CLIP backbone used in our main experiments, we further evaluate it with two representative pre-trained backbones: the visual-only self-supervised DINOv2 ViT-B/14 (Oquab et al., 2023) and the more recent vision-language model SigLIP ViT-B/16 (Zhai et al., 2023). We compare VIS with SimpleCIL and RanPAC using the same backbone and B0 Inc10 protocol on Aircraft and CIFAR100.

As shown in Table 3, VIS consistently achieves the best performance under both DINOv2 and SigLIP, despite noticeable changes in the relative strength of SimpleCIL and RanPAC across different pre-trained representations. Compared with RanPAC, VIS improves average/final-session accuracy by $0.30/0.72$ and $5.08/8.40$ points with DINOv2 on Aircraft and CIFAR100, respectively, and by $6.33/9.09$ and $2.49/3.98$ points with the more recent vision-language model SigLIP. These matched-backbone comparisons show that the effectiveness of VIS generalizes across different pre-training paradigms. In particular, its consistent gains with SigLIP indicate that the visual-space formulation remains effective with a modern vision-language-pretrained representation, further supporting that strong continual classification does not require continued dependence on textual classifier weights.

C.6 FORGETTING ANALYSIS

In addition to average and final-stage accuracy, we further evaluate the forgetting behavior of different methods. Following standard CIL evaluation protocols (Rebuffi et al., 2017; Zhou et al., 2025c), let $\mathcal{A}_{l,b}$ denote the accuracy on task b after learning stage l, and let B be the total number of stages. The forgetting measure after the final stage is defined as:

$$F_B = \frac{1}{B-1} \sum_{b=1}^{B-1} \left(\max_{l \in \{b,\dots,B-1\}} \mathcal{A}_{l,b} - \mathcal{A}_{B,b} \right), \tag{29}$$

where a lower value indicates less forgetting on previously learned tasks.

Table 4: Forgetting measure F_B of different methods under different incremental settings. Lower F_B indicates less forgetting, and the best result in each column is highlighted in bold.

Method	Food		Cars		UCF		SUN	
	B0 Inc10	B50 Inc10	B0 Inc10	B50 Inc10	B0 Inc10	B50 Inc10	B0 Inc30	B150 Inc30
SimpleCIL (Zhou et al., 2025a)	6.22	3.76	5.11	2.51	4.86	2.31	8.12	3.51
ACIL (Zhuang et al., 2022)	4.97	2.94	5.48	2.27	1.17	0.74	8.41	4.52
DualPrompt (Wang et al., 2022b)	24.28	20.77	7.35	4.71	9.26	7.95	23.54	17.63
CODA-Prompt (Smith et al., 2023)	25.82	22.41	6.14	4.59	9.80	9.21	22.71	17.69
RanPAC (McDonnell et al., 2023)	5.39	**2.56**	3.96	**1.71**	1.32	0.98	7.09	**3.34**
RAPF (Huang et al., 2024)	8.16	7.83	28.27	29.30	15.89	20.50	14.58	9.36
CLG-CBM (Yu et al., 2025)	11.79	9.38	6.86	4.86	8.25	6.87	14.62	12.26
PROOF (Zhou et al., 2025c)	16.58	16.94	4.48	2.30	4.22	4.85	18.33	15.71
BOFA (Li et al., 2026)	5.87	4.08	3.94	2.17	4.84	4.13	8.47	4.16
VIS (Ours)	**4.23**	2.68	**3.57**	1.85	**0.72**	**0.38**	**6.97**	3.79

Table 4 shows that VIS consistently achieves low forgetting across datasets and split protocols. It obtains the lowest forgetting in five out of eight settings, including Food B0 Inc10, Cars B0 Inc10, both UCF settings, and SUN B0 Inc30. On the remaining settings, VIS remains close to the best method, with small gaps to RanPAC on Food B50 Inc10, Cars B50 Inc10, and SUN B150 Inc30. These results suggest that recomputing the classifier over all seen classes from additive sufficient statistics effectively preserves old-class information. Compared with optimization-based prompt or adaptation methods, which often suffer larger forgetting under several settings, closed-form statistic-based classifiers generally show more stable old-class retention. Among them, VIS further combines low forgetting with the strong average and final-stage accuracy reported in the main experiments, indicating that its performance gain is not achieved by sacrificing old-class knowledge.

C.7 CONTROLLED EVALUATION OF TEXTUAL INFORMATION

The comparison between ZS-CLIP (Radford et al., 2021) and VIS changes both the use of textual information and the classifier design. To isolate the effect of textual information itself, we therefore construct a controlled variant, denoted as **"VIS + Text"**. It keeps the visual representation, nonlinear feature map, LS-SVM classifier, hyperparameters, and incremental update procedure identical to VIS, while additionally incorporating CLIP textual class prototypes into the same classifier feature space and sufficient statistics. Thus, VIS + Text and VIS differ only in whether textual class information is introduced. For reference, we also report ZS-CLIP results from ENGINE (Zhou et al., 2025b), which uses the same CLIP backbone and incremental protocols.

As shown in Table 5, ZS-CLIP provides a reference for directly using textual embeddings as classifier weights, while the comparison between VIS + Text and VIS more directly isolates the effect of

Table 5: Controlled evaluation of textual information. ZS-CLIP serves as a textual-classifier reference, while VIS + Text and VIS form a controlled pair differing only in the use of textual class prototypes.

Method	Aircraft B0 Inc10		CIFAR100 B0 Inc10	
	$\bar{\mathcal{A}}$	$\mathcal{A}_B$	$\bar{\mathcal{A}}$	$\mathcal{A}_B$
ZS-CLIP (Radford et al., 2021)	26.66	17.22	81.81	71.38
VIS + Text	73.18	64.69	90.56	**85.43**
VIS	**75.21**	**66.46**	**90.64**	85.26

textual information within our framework. Adding textual prototypes provides no consistent benefit: on Aircraft, it decreases average and final-session accuracy by 2.03 and 1.77 points, respectively, while on CIFAR100 the differences are marginal and mixed. These results show that the strong performance of VIS primarily comes from its visual-space formulation and does not rely on additional textual class information.

C.8 TRAINING TIME

We compare the training time of VIS with representative baselines on UCF and Cars in Figure 5c. All training-time results are measured on a single NVIDIA RTX 4090 GPU. VIS is consistently the most efficient method, since only the lightweight visual modules are trained in the base task and later tasks require only sufficient-statistic updates and a closed-form LS-SVM solve. Compared with prompt-based and CLIP-adaptation methods, VIS avoids repeated task-wise optimization, and it is also faster than RanPAC under the same epoch budget.

Table 6: Overhead comparison on **Cars B0 Inc10**. All methods start from the same pre-trained CLIP backbone and are evaluated under the same hardware environment, using a **single RTX 3090 GPU** per method. "Total Time" denotes the end-to-end wall-clock time from the start of continual training to the completion of the final evaluation. "Peak GPU" denotes the peak training GPU memory of the current process, and "Infer." reports the average evaluation latency per test image aggregated over all sessions. Lower is better for all overhead metrics, while higher is better for $\bar{\mathcal{A}}$.

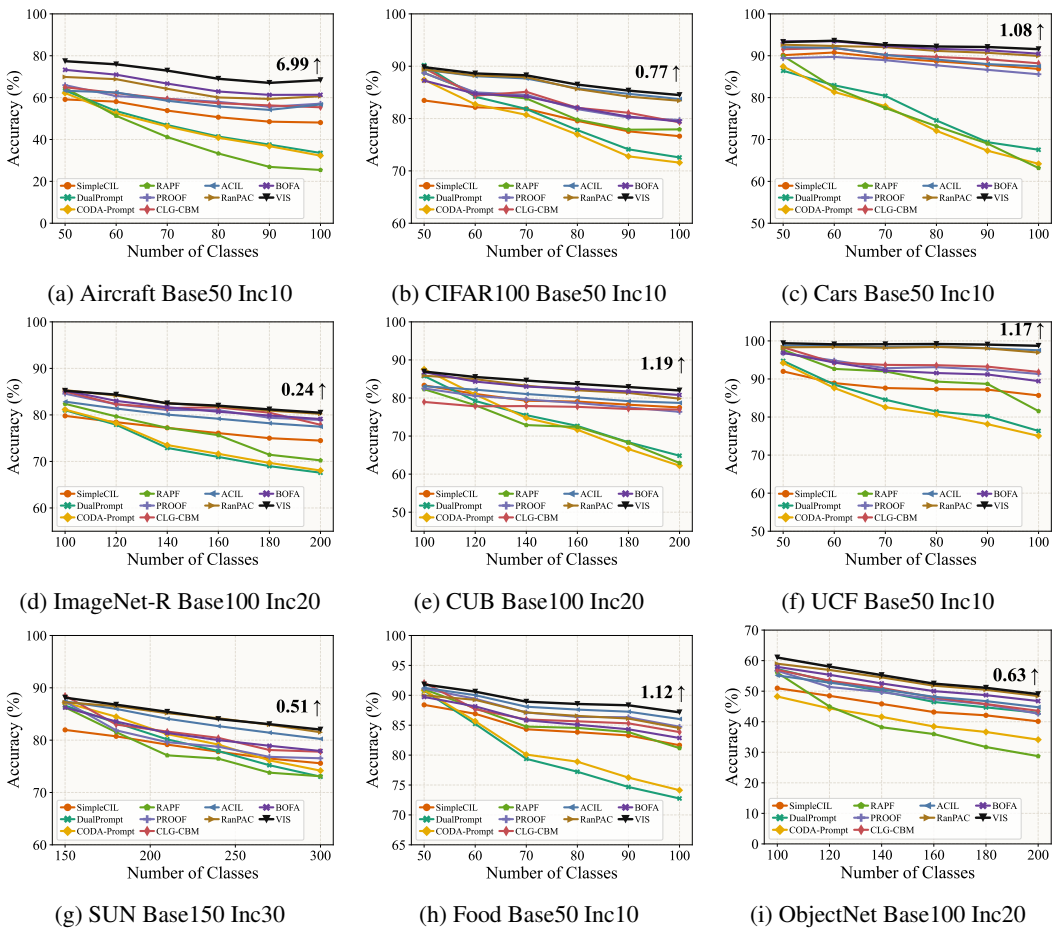
Method	$\bar{\mathcal{A}}$	Total Time (min)↓	Train Time (min)↓	Eval Time (min)↓	Peak GPU (GB)↓	Infer. (ms/img)↓
DualPrompt (Wang et al., 2022b)	63.43	48.90	45.68	3.22	6.95	8.58
CODA-Prompt (Smith et al., 2023)	66.81	53.23	49.64	3.59	15.53	9.58
SimpleCIL (Zhou et al., 2025a)	92.11	**1.50**	**0.37**	**1.13**	**0.92**	3.02
RAPF (Huang et al., 2024)	82.12	19.96	18.62	1.34	6.43	3.59
CLG-CBM (Yu et al., 2025)	93.15	32.90	31.77	**1.13**	1.05	**3.01**
PROOF (Zhou et al., 2025c)	90.44	33.26	30.85	2.41	1.16	6.43
BOFA (Li et al., 2026)	94.26	11.83	9.70	2.13	1.73	5.68
ACIL (Zhuang et al., 2022)	93.08	1.60	0.41	1.20	1.11	3.19
RanPAC (McDonnell et al., 2023)	93.92	6.47	5.16	1.31	5.30	3.49
VIS (Ours)	**94.84**	1.72	0.55	1.17	4.53	3.12

C.9 OVERHEAD ANALYSIS

Table 6 reports the computational overhead on Cars B0 Inc10. All methods use the same pre-trained CLIP backbone, and each method is evaluated on a single NVIDIA RTX 3090 GPU. Among the compared methods, VIS achieves the best average accuracy (94.84%) with competitive overall efficiency.

Training cost. VIS completes the whole continual learning process in 1.72 minutes, which is close to the most efficient baselines SimpleCIL (1.50 min) and ACIL (1.60 min). Its training time is also low (0.55 min), only slightly higher than SimpleCIL (0.37 min) and ACIL (0.41 min). Compared with methods that require heavier task-wise optimization or adaptation, VIS is substantially faster than RanPAC (6.47 min), BOFA (11.83 min), RAPF (19.96 min), CLG-CBM (32.90 min), PROOF (33.26 min), DualPrompt (48.90 min), and CODA-Prompt (53.23 min) in total time. This indicates that the closed-form update in VIS keeps the training overhead low.

Inference cost. VIS requires 3.12 ms per image during evaluation, which is close to efficient baselines such as CLG-CBM (3.01 ms), SimpleCIL (3.02 ms), ACIL (3.19 ms), and RanPAC (3.49 ms). It is also clearly lower than BOFA (5.68 ms), PROOF (6.43 ms), DualPrompt (8.58 ms), and CODA-



(a) Aircraft Base50 Inc10 (b) CIFAR100 Base50 Inc10 (c) Cars Base50 Inc10

(d) ImageNet-R Base100 Inc20 (e) CUB Base100 Inc20 (f) UCF Base50 Inc10

(g) SUN Base150 Inc30 (h) Food Base50 Inc10 (i) ObjectNet Base100 Inc20

Figure 6: Incremental performance of different methods on half-base setting. We report the performance gap after the last incremental stage of VIS and the runner-up method at the end of the line. All methods utilize the same CLIP pre-trained weight.

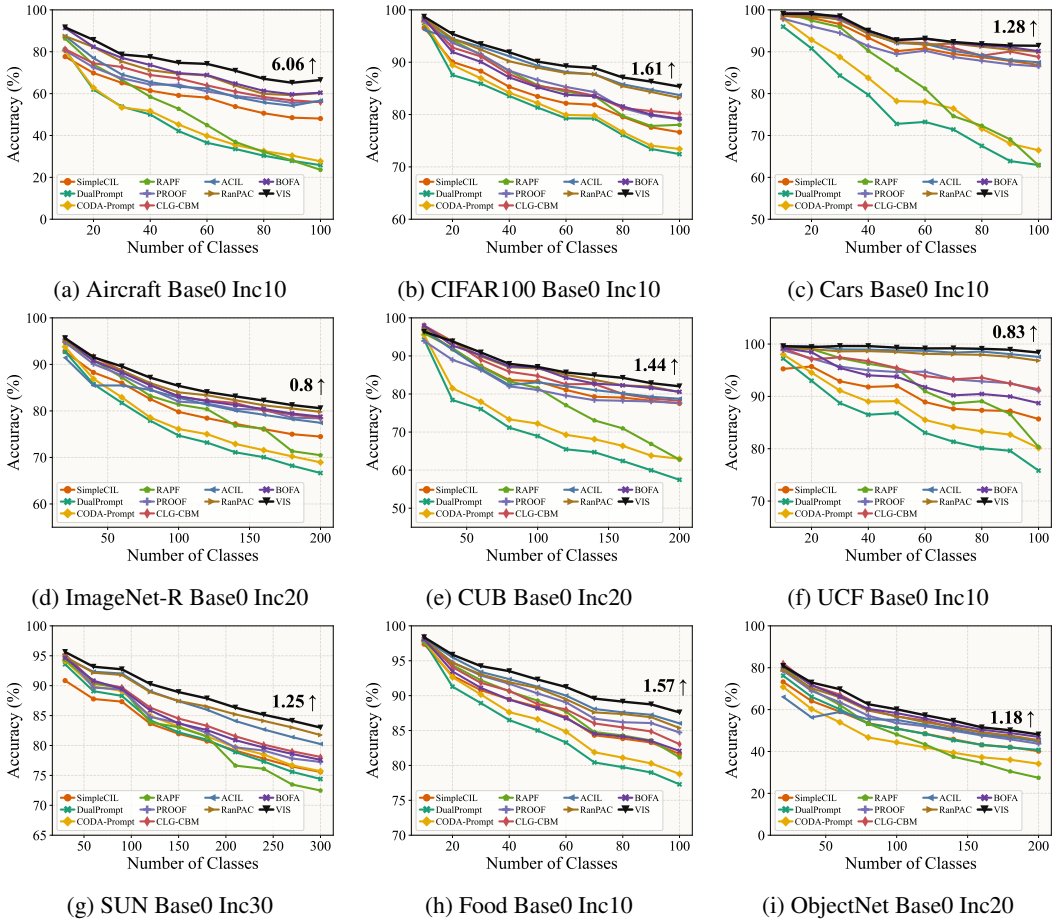
Prompt (9.58 ms). Thus, the improved accuracy of VIS does not introduce a large inference-time burden.

Memory usage. For peak training GPU memory, VIS uses 4.53 GB. This is higher than lightweight methods such as SimpleCIL (0.92 GB), CLG-CBM (1.05 GB), ACIL (1.11 GB), PROOF (1.16 GB), and BOFA (1.73 GB), but lower than RanPAC (5.30 GB), RAPF (6.43 GB), DualPrompt (6.95 GB), and CODA-Prompt (15.53 GB). Therefore, VIS maintains a moderate memory footprint among the compared methods.

Overall, these results indicate that VIS improves accuracy without sacrificing practical efficiency, achieving the best performance among the compared methods while maintaining competitive training time, inference latency, and memory usage.

D FULL RESULTS

In this section, we provide the complete incremental performance curves for all compared methods. While the main paper reports three representative trends, here we include the full set of curves corresponding to Table 1. Specifically, Figure 7 shows the results under the zero-base setting, and Figure 6 reports the half-base setting. Across datasets and split protocols, VIS consistently maintains strong performance and outperforms competing methods in most cases.



(a) Aircraft Base0 Inc10 (b) CIFAR100 Base0 Inc10 (c) Cars Base0 Inc10

(d) ImageNet-R Base0 Inc20 (e) CUB Base0 Inc20 (f) UCF Base0 Inc10

(g) SUN Base0 Inc30 (h) Food Base0 Inc10 (i) ObjectNet Base0 Inc20

Figure 7: Incremental performance of different methods on B0 setting. We report the performance gap after the last incremental stage of VIS and the runner-up method at the end of the line. All methods utilize the same CLIP pre-trained weight.

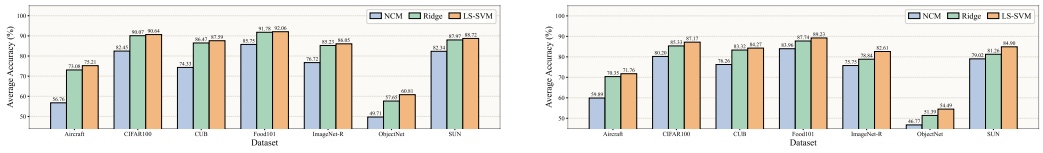
E DETAILS OF COMPARED METHODS

We provide details of the compared methods in the main paper. For fair comparison, all methods are evaluated with the same pre-trained CLIP backbone, and the reproduction is conducted based on the C3Box toolbox (Sun & Zhou, 2026). The methods listed in Table 1 are described as follows:

- **SimpleCIL (Zhou et al., 2025a):** uses the frozen CLIP visual encoder as a general-purpose feature extractor and removes the language branch during evaluation. For each newly observed class, it computes visual prototypes from the extracted features and performs prediction with a cosine classifier. This baseline reflects the strength of frozen CLIP visual representations without any task-wise parameter update.
- **ACIL (Zhuang et al., 2022):** ACIL learns an analytic classifier for class-incremental learning by updating sufficient statistics and solving the classifier in closed form. The original ACIL uses a separately trained visual backbone, whereas we adapt it to our CLIP-based setting by using the same frozen CLIP visual backbone as the feature extractor. This removes backbone differences and makes the comparison focus on the analytic classification mechanism.
- **RanPAC (McDonnell et al., 2023):** RanPAC combines pre-trained representations with random projections and parameter-efficient adaptation for continual learning. For fair comparison, we evaluate it under the same CLIP backbone and the unified training budget used in our benchmark. This allows us to compare its PETL-based random-projection pipeline

with our frozen-backbone visual-only statistic-based classifier under a shared pre-trained representation.

- **DualPrompt (Wang et al., 2022b):** is a prompt-based continual learning method that introduces both general prompts and expert prompts on top of a frozen pre-trained backbone. It selects task-adaptive prompts from a prompt pool to guide the visual representation, and in our comparison it operates on the visual branch of CLIP.
- **CODA-Prompt (Smith et al., 2023):** extends prompt-based adaptation by replacing hard prompt selection with attention-based prompt recombination. Instead of choosing a fixed prompt for each instance, it learns to compose prompts dynamically, while still adapting the frozen CLIP visual branch.
- **RAPF (Huang et al., 2024):** is a CLIP-based CIL method that updates the model with adaptive representation adjustment and parameter fusion. It introduces class-separation constraints and decomposed fusion to incorporate new-task information while mitigating interference with previously learned knowledge.
- **PROOF (Zhou et al., 2025c):** improves continual learning for vision-language models by introducing expandable projection layers and a cross-modal fusion mechanism. It leverages both visual and textual prototypes and refines their interaction to enhance incremental recognition.
- **CLG-CBM (Yu et al., 2025):** builds a language-guided concept bottleneck model for interpretable continual learning. By aligning CLIP representations with semantic concepts, it aims to learn concepts that are understandable and transferable across tasks.
- **BOFA (Li et al., 2026):** proposes bridge-layer orthogonal low-rank fusion for CLIP-based CIL. It uses lightweight low-rank updates at intermediate layers and imposes orthogonality constraints to reduce interference between old and new tasks during incremental adaptation.



(a) B0 protocol

(b) B-half protocol

Figure 8: Classifier-choice ablation under fixed kernel-induced visual features. All methods use the same enhanced representation and kernel feature map, and differ only in the final classifier. NCM uses class means, Ridge uses one-hot least-squares targets, and LS-SVM uses one-vs-all ± 1 targets. Ridge and LS-SVM both outperform NCM, while LS-SVM achieves comparable or better performance than Ridge in most settings, supporting our SVM-style positive-versus-negative classifier formulation.

F Classifier Choice under Fixed Features

We further study the effect of the final classifier while keeping the feature pipeline fixed. All variants use the same enhanced visual representation and the same kernel-induced feature map, and differ only in the final classification rule. We compare three classifiers: NCM, which assigns samples to the nearest class mean; Ridge, which solves an ordinary least-squares classifier with one-hot targets; and LS-SVM, which uses one-vs-all ± 1 targets.

Figure 8 reports the results under both B0 and B-half protocols. NCM is consistently worse than Ridge and LS-SVM across datasets, showing that class centroids alone are insufficient even in the kernel-induced visual feature space. Both Ridge and LS-SVM benefit from closed-form least-squares classification on the same features, but LS-SVM achieves comparable or better performance than Ridge in most settings. This suggests that the SVM-style one-vs-all formulation is at least as effective as one-hot least-squares classification under the same feature pipeline, while providing a more suitable formulation for our incremental kernelized classifier. Unlike one-hot Ridge,

where non-target classes are assigned zero targets, LS-SVM uses one-vs-all ± 1 coding and explicitly treats every non-ground-truth class as a negative class. This positive-versus-negative supervision matches the SVM-style decision formulation and is consistent with our sufficient-statistics update: when new classes arrive, previous samples can be incorporated as negative evidence for the new-class classifiers through the stored feature-sum statistic, without revisiting the original data. Moreover, SVMs are classical kernel machines: their decision functions can be formulated in a feature space induced by a kernel, and LS-SVM inherits this kernelized formulation while yielding a closed-form least-squares solution. This naturally aligns with our use of a kernel-induced visual feature space. Therefore, we adopt LS-SVM as a principled closed-form classifier that retains the analytic-update advantage of Ridge while providing an SVM-style discriminative interpretation for incremental learning.

G Comparison with Related Closed-Form CIL Methods

ACIL (Zhuang et al., 2022) and RanPAC (McDonnell et al., 2023) are closely related to VIS in that they construct incremental classifiers from fixed or expanded feature representations without repeatedly optimizing the classifier through gradient descent. Nevertheless, VIS differs from them in its motivation, representation construction, and classifier update.

Motivation: ACIL is primarily motivated by preserving historical knowledge without replay, and derives a recursive least-squares update that reproduces its joint-learning solution. RanPAC instead focuses on exploiting pre-trained representations for continual learning while avoiding forgetting from repeated parameter updates. In contrast, VIS starts from a CLIP-specific observation: textual classifier weights can be misaligned with visual class distributions and lead to less favorable classifier optimization. Our goal is therefore to examine whether the textual branch is necessary for CLIP-based CIL and to construct the incremental classifier entirely in the visual space.

Representation Construction: Both ACIL and RanPAC increase classifier capacity through randomized feature expansion. ACIL applies a randomly initialized feature-expansion layer with nonlinear activation to the extracted representation, whereas RanPAC combines pre-trained features with an optional first-session PETL adaptation followed by a fixed nonlinear random projection. VIS differs primarily in how the representation before this expansion is constructed: it learns a residual correction from multi-level CLIP visual features using only base-session data, thereby adapting the final visual representation to the downstream distribution before applying the fixed nonlinear feature map.

Classifier Update: The main difference lies in the target formulation and its consequence for class expansion. ACIL uses one-hot labels and recursively updates a regularized least-squares classifier, while RanPAC accumulates a Gram matrix and class prototypes, which likewise correspond to a regularized least-squares solution with one-hot targets. For these formulations, historical samples have zero targets for newly introduced output dimensions. In contrast, VIS adopts a one-vs-all LS-SVM with ± 1 targets. When a new class arrives, every historical sample should contribute a target of -1 to its classifier; VIS therefore maintains the additional feature-sum statistic $\mathbf{s}_t$ to recover this contribution without revisiting previous data, and recomputes the classifiers for all seen classes from $(G_t, Q_t, \mathbf{s}_t)$. The effect of this ± 1 formulation relative to one-hot Ridge regression is further isolated in Section F.

Empirical Comparison: Under the same CLIP backbone and continual-learning protocols, VIS consistently outperforms both ACIL and RanPAC in average and final-session accuracy across the benchmark settings in Table 1. Together with the controlled Ridge-versus-LS-SVM comparison in Section F, these results indicate that the gains of VIS do not simply come from randomized feature expansion or least-squares-style classifier updates, but from the combination of task-adaptive visual representation construction and the proposed one-vs-all LS-SVM formulation.

H Supplementary Preliminary Experiments

We provide additional preliminary experiments to support the diagnosis in the main paper. These results further examine the relationship between image-text modality gap, text-head degradation, and the optimization behavior of textual versus image-prototype initialization.

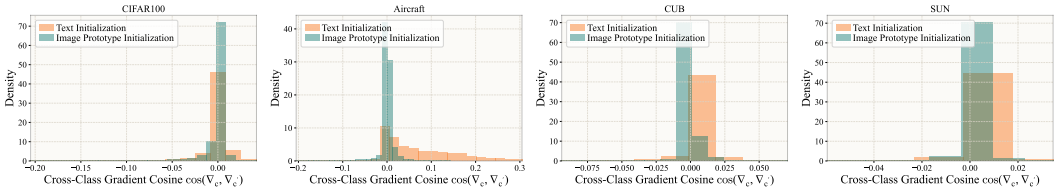


Figure 9: Cross-class gradient cosine distributions under textual initialization and image-prototype initialization on representative datasets. We compute pairwise cosine similarities $\cos(\nabla_c, \nabla_{c'})$ between class-wise gradient directions. Compared with textual initialization, image-prototype initialization generally produces distributions that are more concentrated around zero, indicating more balanced class-wise optimization directions.

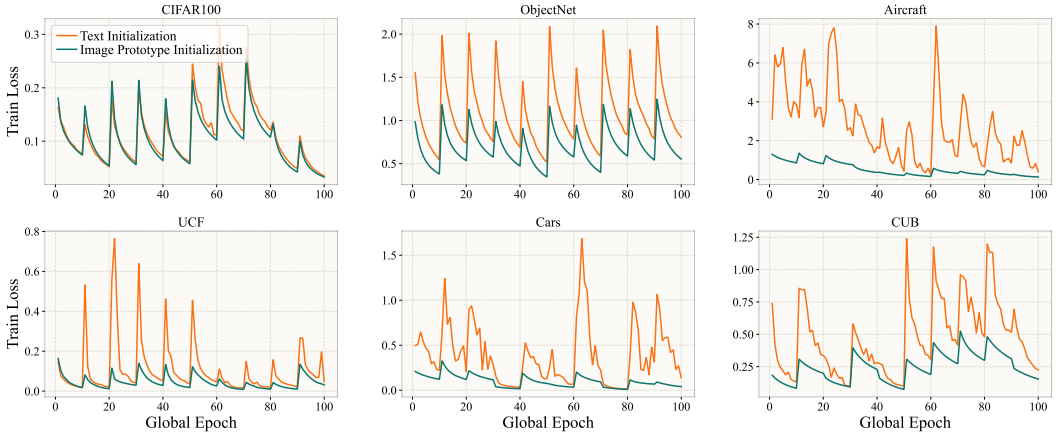


Figure 10: Training-loss curves under textual initialization and image-prototype initialization on the same six representative datasets as Figure 12. Textual initialization often leads to larger loss values and stronger loss fluctuations, while image-prototype initialization produces lower or smoother loss curves. This suggests that visual prototypes provide a more stable initialization for classifier optimization.

Additional modality-gap analysis. Figure 11 extends the class-wise gap analysis to nine datasets. For each class c, we compute the modality gap g_c between its image prototype and textual embedding, and measure the class-wise accuracy improvement Δ_c of image-prototype initialization over textual initialization. Each point in the figure corresponds to one class. Across most datasets, g_c and Δ_c show a positive Pearson correlation, although the correlation strength varies across benchmarks. This indicates that classes with larger image-text mismatch tend to suffer more from text-based classifier weights, and therefore benefit more from visual prototypes.

Additional optimization dynamics. We further compare the training dynamics of textual initialization and image-prototype initialization on six representative datasets. Figure 12 reports the seen-class accuracy over global epochs, and Figure 10 reports the corresponding training loss. Compared with textual initialization, image-prototype initialization generally leads to higher seen-class accuracy and lower or smoother training loss. The difference is especially clear on datasets such as ObjectNet, Aircraft, UCF, Cars, and CUB, where textual initialization results in larger loss spikes or worse seen-class accuracy. These results provide additional evidence that textual embeddings may provide a less favorable starting point for optimizing the classifier.

Cross-class gradient statistics. To further inspect the optimization behavior, we analyze cross-class gradient cosine similarity on representative datasets. For each class c, we compute the gradient direction ∇_c induced by samples from that class, and then measure pairwise cosine similarities $\cos(\nabla_c, \nabla_{c'})$ between different classes. This statistic reflects how aligned or conflicting the class-wise optimization directions are. A more concentrated distribution around zero indicates less biased cross-class coupling and more balanced optimization directions. As shown in Figure 9, image-

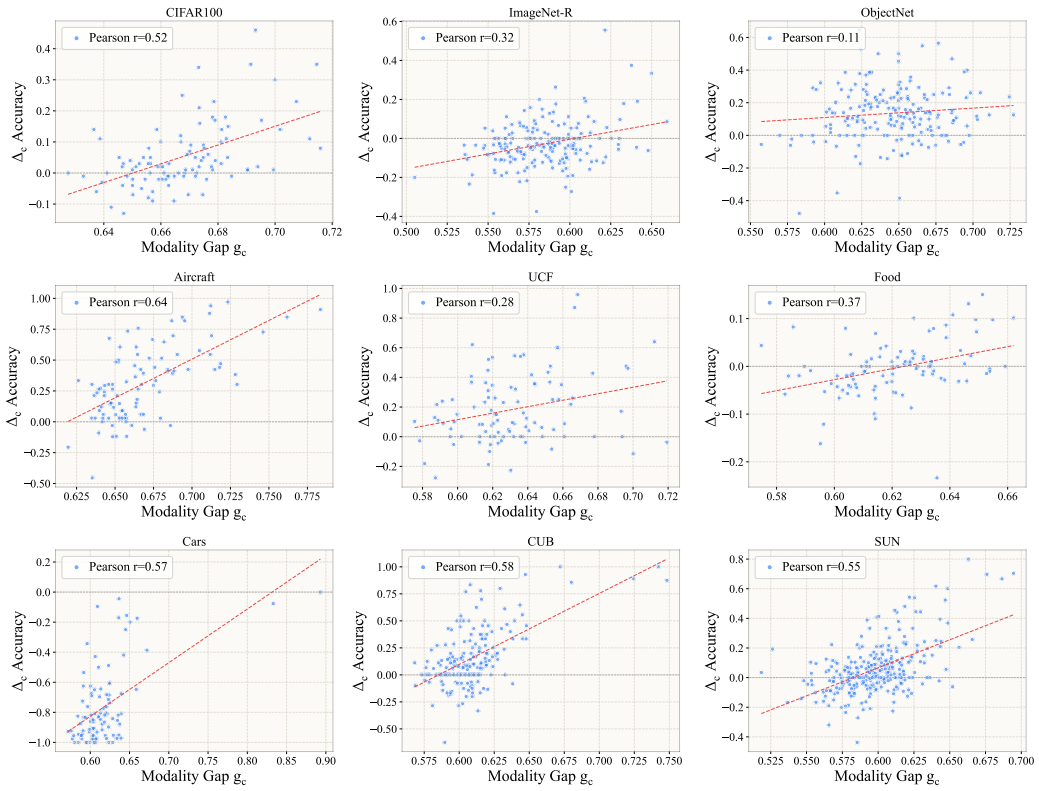


Figure 11: Additional class-wise modality-gap analysis on nine datasets. Each point denotes one class. The x-axis is the class-level modality gap g_c between the image prototype and textual embedding, and the y-axis is the class-wise accuracy improvement Δ_c of image-prototype initialization over textual initialization. The dashed red line shows the linear fit, and the Pearson correlation coefficient is reported in each subplot. Most datasets show a positive correlation, suggesting that classes with larger image-text gaps tend to benefit more from image-prototype initialization.

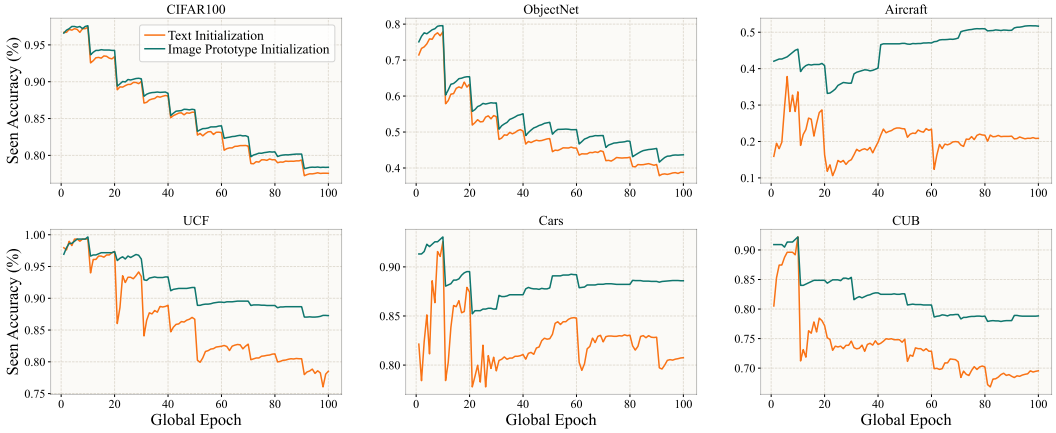


Figure 12: Seen-class accuracy curves under textual initialization and image-prototype initialization on six representative datasets. The x-axis denotes global training epochs, and the y-axis denotes the accuracy over all seen classes. Image-prototype initialization generally yields higher and more stable seen-class accuracy, indicating a more favorable optimization trajectory.

prototype initialization generally produces more concentrated gradient-cosine distributions, whereas textual initialization often shows broader distributions or heavier tails. This provides another view of why textual initialization can be less favorable for classifier optimization.

Additional t-SNE visualization. Figure 13 provides additional t-SNE visualizations of image samples, image prototypes, and textual embeddings in the shared CLIP embedding space. Across datasets, textual embeddings often occupy regions that are clearly separated from the corresponding image distributions, while image prototypes lie much closer to image samples. This qualitative observation is consistent with the modality-gap analysis above and supports the motivation of constructing the classifier in the visual space.

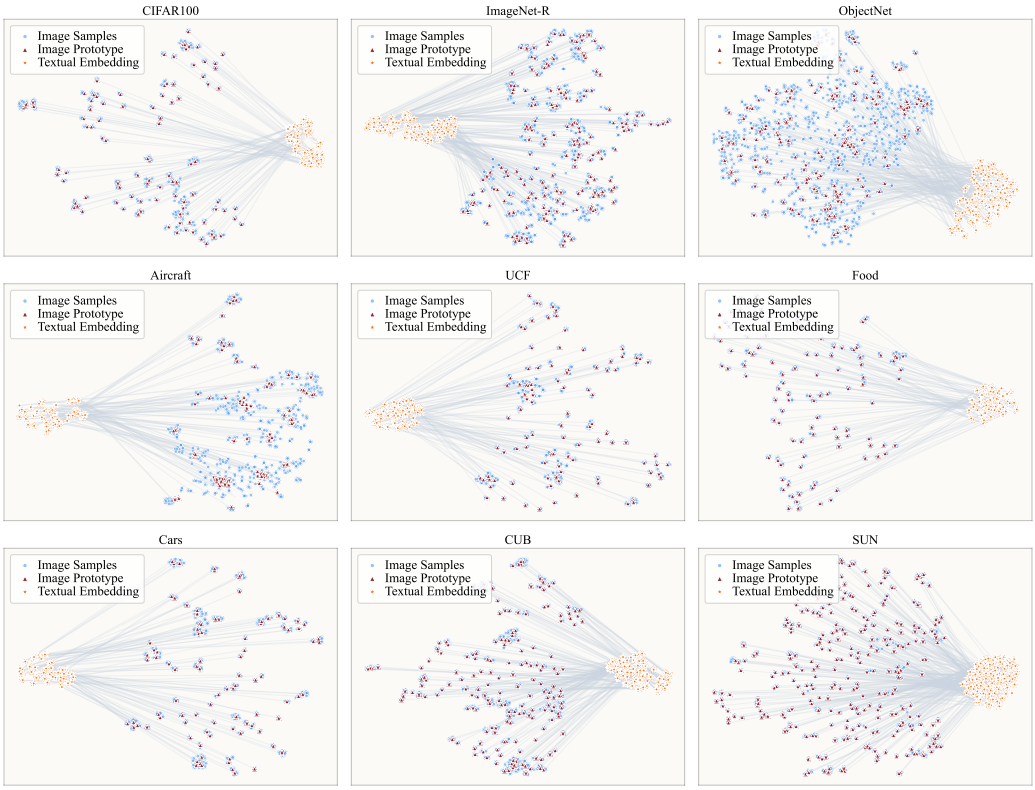


Figure 13: Additional t-SNE visualizations of image samples, image prototypes, and textual embeddings on nine datasets. Blue dots denote image samples, red triangles denote image prototypes, and orange stars denote textual embeddings. Textual embeddings often lie in regions separated from the corresponding image distributions, while image prototypes remain close to image samples. This illustrates the image-text modality gap in the shared CLIP embedding space.