LACE: LIGHTWEIGHT ATTRIBUTION-GUIDED CONCEPT EVOLUTION FOR CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

We study concept-bank growth in exemplar-free class-incremental learning. LACE combines task-local concept ranking, an explicit retention budget, and prototype augmentation for old-class training support. We derive the nonnegative form of the specified mean-gradient Concept Attribution (CA) rule and a finite-step loss-decomposition bound. An optional Concept Verification (CV) heuristic compares candidate budgets using a local complexity correction. Archived ImageNet-100 ablations report 900 rather than 1,000 retained coordinates at average accuracy of $93.35 \pm 0.46\%$ versus $93.78 \pm 0.35\%$ without pruning, with prototype augmentation having the strongest accuracy effect. Source inspection identifies a legacy samplewise CA implementation and test-based selection, so these measurements describe the historical pipeline rather than validate the analyzed rule or an independent held-out protocol. The resulting formulation separates coordinate ranking, budget control, and old-class support, and makes the remaining implementation-verification requirements explicit.

1 INTRODUCTION

Continual Learning (CL) learns from sequential tasks while limiting catastrophic forgetting (Hadsell et al., 2020; Kirkpatrick et al., 2017). Beyond accuracy, an auditable learner should expose how its decision basis changes as classes arrive. Concept Bottleneck Models (CBMs) provide an explicit intermediate representation (Koh et al., 2020); language-guided variants use language models and CLIP to propose and align named concepts (Brown et al., 2020; Radford et al., 2021; Yuksekgonul et al., 2022). In continual CBMs, these coordinates can be retained across tasks and inspected alongside newly introduced concepts (Yu et al., 2025).

Expansion alone leaves a practical problem: the number of concepts to inspect grows with the task stream. A smaller bank reduces this burden, but neither semantic quality nor reduced forgetting follows from cardinality alone. We therefore study an explicit compactness–accuracy trade-off while preserving the identity and selection history of each coordinate.

LACE (Lightweight Attribution-guided Concept Evolution) separates these roles. Prototype augmentation stabilizes exemplar-free training. CA assigns task-loss scores to the current concept block; fixed or adaptively selected budgets then determine which coordinates survive. A concept can be meaningful but weakly predictive for the current loss, so selection records provide model-level auditability rather than a human semantic-quality certificate. Our principal evidence is the fixed-budget comparison; optional CV diagnostics evaluate a separate budget-selection component.

Contributions. We introduce a concept-control loop for continual language-guided CBMs, specify its data-dependent ranking with a bounded discrete loss decomposition, and expose the accuracy–bank-size trade-off through component ablations. The contribution is methodological: the mathematics explains the score and the limits of a local budget heuristic, without asserting globally optimal pruning or a forgetting bound.

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.

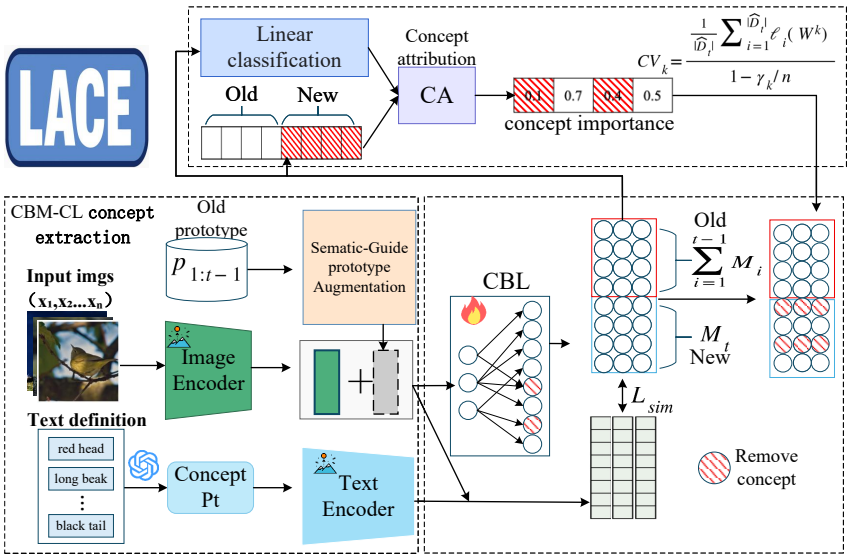


Figure 1: LACE overview. Hatching marks pruned current-task concepts; old concept identities are retained, while classifier weights may still be updated. The CV formula denotes the optional budget heuristic; the primary comparisons use fixed budgets. Prototype augmentation supplies old-class training support.

2 Related Work

2.1 Continual Learning

Regularization methods constrain parameter drift (LwF, EWC), replay methods revisit earlier information (iCaRL), and expanding architectures isolate task knowledge (Progressive Neural Networks) (Li & Hoiem, 2017; Kirkpatrick et al., 2017; Rebuffi et al., 2017; Rusu et al., 2016). Prompt methods such as L2P, DualPrompt, and CODA-Prompt adapt pretrained models without storing old images (Wang et al., 2022b;a; Smith et al., 2023). ICICLE connects class-incremental learning with interpretable prototypes (Rymarczyk et al., 2023). LACE instead controls the evolving language-named coordinate bank of a continual CBM.

2.2 Concept Bottleneck Models (CBMs), Concept Cardinality, and Pruning

CBMs predict through explicit concepts (Koh et al., 2020). Post-hoc CBMs, LaBo, and label-free CBMs extend this interface using learned or language-derived concept spaces (Yuksekgonul et al., 2022; Yang et al., 2023; Oikarinen et al., 2023). CLG-CBM carries this construction into continual learning (Yu et al., 2025) and is our closest backbone-level comparator. ProtoPNet and ProtoTree offer related examples of inspection through visual prototypes (Chen et al., 2019; Nauta et al., 2021). LACE adds task-local retention control after concept expansion. Cardinality is an inspection-cost measure, not evidence that the retained concepts are less redundant, better grounded, or easier for people to use.

2.3 Attribution and Concept Attribution (CA)

Integrated Gradients (IG) formalizes input attribution using Sensitivity and Implementation Invariance (Sundararajan et al., 2017). TCAV measures concept-direction sensitivity, while ConceptSHAP allocates importance across concept subsets (Kim et al., 2018; Yeh et al., 2020). Grad-CAM and RISE provide spatial and masking-based explanations (Selvaraju et al., 2017; Petsiuk et al., 2018). Global concept attribution also predates LACE: Attacking for Interpretability evaluates user-defined concepts

over sample groups (Wu et al., 2020). Inspired by gradient-path methods including MFABA (Zhu et al., 2024), our CA accumulates task-average loss gradients in a fixed concept coordinate system. Its role is to rank newly introduced coordinates for retention, rather than to claim a new general attribution axiom or cross-task calibration.

2.4 Selecting the Pruning Budget (CV)

Classical cross-validation motivates comparison of candidate model sizes (Stone, 1974; Arlot & Celisse, 2010). Our Concept Verification (CV) is a locally motivated complexity-corrected training-loss heuristic for comparing pruning budgets without a separate validation set. It is not a prior concept-verification algorithm or exact leave-one-out cross-validation. CA determines the ordering; a fixed ratio or CV determines the retained count.

3 Method

3.1 Problem Definition

In the class-incremental learning (CIL) setting, a model is required to learn new classes sequentially over a series of tasks while retaining knowledge from earlier tasks (Yu et al., 2025). Let the task index set be $\mathcal{T} = \{1, 2, \ldots, T\}$. For task t, the dataset is

$$\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^{N_t}, \quad x_i \in \mathcal{X}_t,\ y_i \in \mathcal{Y}_t, \tag{1}$$

where N_t denotes the number of samples in task t and $\mathcal{Y}_t$ is the label set for task t. The label sets across different tasks are disjoint: $\mathcal{Y}_i \cap \mathcal{Y}_j = \varnothing \quad (\forall\, i \neq j)$. After completing t tasks, the cumulative label space is $\mathcal{Y}_{1:t} = \mathcal{Y}_1 \cup \mathcal{Y}_2 \cup \cdots \cup \mathcal{Y}_t$, with the total number of classes given by $|\mathcal{Y}_{1:T}| = C$. During training on task t, the learner has access only to $\mathcal{D}_t$, whereas at inference time it must predict over $\mathcal{Y}_{1:t}$.

For representation learning, we adopt the CLIP framework (Radford et al., 2021) and introduce an image encoder $E_I(\cdot)$ and a text encoder $E_T(\cdot)$. Both map images and texts, respectively, into a shared semantic space with output dimensionality d.

We treat each bank as an ordered concatenation of task-indexed coordinates: $\cup$ below denotes concatenation, not deduplication of repeated strings. We use explicit pre- and post-pruning notation. Let $\mathcal{P}_t^{\text{new}}$ be the concepts generated for task t, $\hat{\mathcal{P}}_{1:t-1}$ the retained concepts from previous tasks, and $\hat{\mathcal{P}}_{1:t}^{\text{post}}$ the retained pool after pruning task t. Before pruning task t, the concept dimension is $m_t^{\text{pre}} = \sum_{\tau=1}^{t-1} \hat{M}_\tau + M_t$; after pruning it is $m_t^{\text{post}} = \sum_{\tau=1}^{t} \hat{M}_\tau$. Thus an empty initial bank with M candidates per task and exactly $(1-r)M$ retained has $m_t^{\text{post}} = t(1-r)M$ (with integer rounding applied per task if needed). At $M = 100, r = 0.1$, this is $90, 180, \ldots, 900$ for ten tasks. This is a protocol-derived coordinate count, not a measured semantic-reuse or redundancy curve; fixed fractional pruning still gives linear growth.

3.2 Architecture of CBM-CL

This subsection presents the CBM-CL pipeline, which proceeds in three stages: **concept selection**, **concept alignment**, and **prototype augmentation**.

Concept generation and selection. For task t, we employ an LLM to propose semantic concepts for each class in $\mathcal{Y}_t$ (Oikarinen et al., 2023). This yields a task-specific concept pool $\mathcal{P}_t^{\text{new}} = \{c_1^t, c_2^t, \ldots, c_{M_t}^t\}$, where M_t is the number of newly generated concepts for task t. Before pruning, the expanded concept pool is $\hat{\mathcal{P}}_{1:t}^{\text{pre}} = \hat{\mathcal{P}}_{1:t-1} \cup \mathcal{P}_t^{\text{new}}$.

To leverage these concepts, we encode them with CLIP's text encoder $E_T(\cdot)$ to obtain a bank of concept vectors $\mathbf{C}_t = \{E_T(c_1^t), E_T(c_2^t), \ldots, E_T(c_{M_t}^t)\}, \quad \mathbf{C}_t \in \mathbb{R}^{M_t \times d}$, with embedding dimension $d = 512$. The pre-pruning concept-vector bank is $\hat{\mathbf{C}}_{1:t}^{\text{pre}} = \hat{\mathbf{C}}_{1:t-1} \cup \mathbf{C}_t, \qquad \hat{\mathbf{C}}_{1:t}^{\text{pre}} \in \mathbb{R}^{m_t^{\text{pre}} \times d}$.

Concept alignment. The Concept Bottleneck Layer (CBL) maps image features to concepts. Write $\mathbf{W}_c^t \equiv \mathbf{W}_c^{t,\text{pre}} \in \mathbb{R}^{m_t^{\text{pre}} \times d}$. If $S_t \in \{0,1\}^{m_t^{\text{post}} \times m_t^{\text{pre}}}$ selects the retained rows in their original order,

then

$$\mathbf{W}_c^{t,\text{post}} = S_t \mathbf{W}_c^{t,\text{pre}}, \qquad \mathbf{W}_c^{t+1,\text{pre}} = \begin{bmatrix} \mathbf{W}_c^{t,\text{post}} \\ w^{\text{init}} \end{bmatrix},$$

where $w^{\text{init}} \in \mathbb{R}^{M_{t+1} \times d}$ initializes the new rows. Hence $m_{t+1}^{\text{pre}} = m_t^{\text{post}} + M_{t+1}$; removed coordinates are not carried forward in this mathematical representation.

We optimize $(\mathbf{W}_c^t)^\top$ with the similarity loss. For each image x_i, let

$$\hat{s}_i = E_I(x_i)(\mathbf{W}_c^t)^\top \in \mathbb{R}^{m_t^{\text{pre}}}, \qquad s_i^{\text{clip}} = E_I(x_i)(\hat{\mathbf{C}}_{1:t}^{\text{pre}})^\top \in \mathbb{R}^{m_t^{\text{pre}}}.$$

We define

$$\mathcal{L}_{\text{sim}} = 1 - \cos\left((\hat{s}_i)^3,\ \left(s_i^{\text{clip}}\right)^3 \right), \tag{2}$$

which aligns the learned concept scores with CLIP's concept space. Using raw scores often yields many mid-range similarities (0.3–0.6), producing insufficient sparsity and salience; the cubic sharpening emphasizes high scores and suppresses low scores to obtain a peakier distribution (Yu et al., 2025).

We further introduce a linear head from concepts to classes with parameters

$$\begin{aligned} &\mathbf{W}_l^t \in \mathbb{R}^{m_t^{\text{pre}} \times |\mathcal{Y}_{1:t}|}, \\ &\mathbf{W}_l^{t+1} \in \mathbb{R}^{m_{t+1}^{\text{pre}} \times |\mathcal{Y}_{1:t+1}|}. \end{aligned} \tag{3}$$

Write $\mathbf{W}_l^t \equiv \mathbf{W}_l^{t,\text{pre}}$ and $C_t = |\mathcal{Y}_{1:t}|$. Pruning gives $\mathbf{W}_l^{t,\text{post}} = S_t \mathbf{W}_l^{t,\text{pre}} \in \mathbb{R}^{m_t^{\text{post}} \times C_t}$. The next classifier expands from this matrix with M_{t+1} new rows and $C_{t+1} - C_t$ new columns. Logits use all classes seen so far. For a sample x_i, the prediction is $\hat{y}_i = \hat{s}_i \, \mathbf{W}_l^t$.

We use logit-space cross-entropy $\ell(z, y) = -z_y + \log \sum_c \exp z_c$ throughout, so $\mathcal{L}_{\text{ce}}(z, y) \equiv \ell(z, y)$ applies softmax exactly once. The training objective is $\mathcal{L} = \mathcal{L}_{\text{ce}}(\hat{y}, y) + \lambda \mathcal{L}_{\text{sim}} + \sigma \mathcal{L}_{\text{sparse}}$, where λ, σ weight the terms and $\mathcal{L}_{\text{sparse}}$ denotes the archived classifier-regularization term (Yu et al., 2025). Its exact form and coefficients, and which historical parameter blocks are updated, need run-specific configuration; they are not supplied by this abstract objective.

Prototype augmentation. To mitigate catastrophic forgetting, we augment training with class prototypes. For each class in the current task,

$$P_i = \frac{1}{|\mathcal{D}_t^i|} \sum_{x \in \mathcal{D}_t^i} E_I(x), \qquad i \in \mathcal{Y}_t, \tag{4}$$

where $\mathcal{D}_t^i$ is the subset of $\mathcal{D}_t$ with label i. Because prototypes are defined in CLIP's image space, we cache the prototype of each class when it is last updated. For a past class $j \in \mathcal{Y}_{1:(t-1)}$, let P_j^{old} denote this cached prototype. We match historical classes to current classes using their text embeddings and current prototypes: $h_j = \arg\max_{i \in \mathcal{Y}_t} \langle E_T(y_j),\, P_i \rangle$.

Using images from class h_j, we construct pseudo-features for the old class j. Let $\mathbf{V}^{h_j} = \{E_I(x) \mid x \in \mathcal{D}_t^{h_j}\}$, then the pseudo-features are

$$\tilde{\mathbf{V}}_{\text{pseudo}}^j = \{P_j^{\text{old}} + (v - P_{h_j}) : v \in \mathbf{V}^{h_j}\}, \tag{5}$$

where P_{h_j} is the prototype of the matched current class h_j. Each current-task feature is translated from its class prototype to the cached old-class prototype, so all pseudo-items remain in the CLIP image-feature space.

We mix these pseudo-features with the current data to define an augmented set

$$\mathcal{D}_t^a = \{(v, j) \mid v \in \tilde{\mathbf{V}}_{\text{pseudo}}^j,\ j \in \mathcal{Y}_{1:(t-1)}\}, \tag{6}$$

and the final training set is

$$\hat{\mathcal{D}}_t = \{(E_I(x), y) : (x, y) \in \mathcal{D}_t\} \cup \mathcal{D}_t^a. \tag{7}$$

Thus all items in $\hat{\mathcal{D}}_t$ are feature-label pairs and can be used directly by the concept layer and linear classifier.

Algorithm 1 LACE task loop

1: **for** task $t = 1, \ldots, T$ **do**
2: Generate $\mathcal{P}_t^{\text{new}}$ and expand the retained concept bank to $\hat{\mathcal{P}}_{1:t}^{\text{pre}}$.
3: Expand $\mathbf{W}_c^t$ and $\mathbf{W}_l^t$ to match m_t^{pre} and $\mathcal{Y}_{1:t}$.
4: Build $\hat{\mathcal{D}}_t$ by concatenating current CLIP features with cached-prototype pseudo-features for old classes.
5: Train the concept-alignment layer and sparse classifier on $\hat{\mathcal{D}}_t$.
6: Compute CA scores A_j^t for the current-task concept block.
7: Set $\hat{M}_t$ by a fixed budget, or compare candidate budgets with optional CV.
8: Prune only current-task concepts, cache updated class prototypes, and set $\hat{\mathcal{P}}_{1:t}^{\text{post}}$.
9: **end for**

3.3 Concept Importance Estimation

We rank and prune only the newly introduced concept coordinates $\mathcal{P}_t^{\text{new}}$, retaining the historical bank $\hat{\mathcal{P}}_{1:t-1}$. This gives a recoverable task-local decision record and avoids reassessing old coordinates without old-task data. It does not guarantee that the historical bank remains semantically optimal or that pruning improves forgetting.

Inspired by exploration-based attribution (see (Zhu et al., 2024)), we *probe within the concept space* to quantify each concept's impact on the current model loss, while using the two central attribution axioms—*Sensitivity* and *Implementation Invariance* (see (Sundararajan et al., 2017))—as design guidance rather than claiming full axiom equivalence.

During CA computation, we make four assumptions: (A1) $\mathcal{L}(E)$ is differentiable in a neighborhood of the exploration path; (A2) model parameters are frozen while CA is computed; (A3) the reference step size $\eta_{\text{CA}} \geq 0$ and step count G_{CA} are fixed during a score calculation; and (A4) CA ranks only the current-task concept block while previously retained concepts remain fixed.

At task t, denote the learned concept-mapping weights by $\mathbf{W}_c^t$ and the classification-head weights by $\mathbf{W}_l^t$. For $(v_i, y_i) \in \hat{\mathcal{D}}_t$ (write $n_t = |\hat{\mathcal{D}}_t|$), define $\bar{E}_i = v_i\,(\mathbf{W}_c^t)^\top, \mathcal{L}(\bar{E}_i, y_i) = \ell(\bar{E}_i \mathbf{W}_l^t, y_i)$, and abbreviate $\mathcal{L}(E_i) \equiv \mathcal{L}(\bar{E}_i, y_i)$ when only the current-task block is perturbed and historical coordinates are held fixed. During CA, E_i denotes this current-task block, and $m_t = M_t$ is its dimensionality. At exploration step $g \in \{0, \ldots, G_{\text{CA}}\}$, the concept representation for sample i is $E_i^g \in \mathbb{R}^{m_t}$, with increment $\Delta E_i^g = E_i^{g+1} - E_i^g$. Initialize E_i^0 as the current-task block of $\bar{E}_i$ and set $\text{sign}(0) = 0$. The reference ranking breaks equal scores by original coordinate index. These deterministic conventions specify the analyzed rule, not an audited legacy tie policy.

The specified default update uses the dataset-averaged gradient, without sample-specific clipping or projection:

$$E_i^{g+1} = E_i^g + \eta_{\text{CA}} \cdot \text{sign}\left(\frac{1}{n_t}\sum_{i'=1}^{n_t} \nabla_E \mathcal{L}\big(E_{i'}^g\big)\right), \quad g = 0, \ldots, G_{\text{CA}} - 1. \tag{8}$$

The dataset-level attribution is then

$$\begin{aligned} A^t &:= \frac{1}{n_t}\sum_{i=1}^{n_t}\left[\mathcal{L}(E_i^{G_{\text{CA}}}) - \mathcal{L}(E_i^0)\right] \\ &\approx \frac{1}{n_t}\sum_{i=1}^{n_t}\sum_{g=0}^{G_{\text{CA}}-1}\left\langle \nabla_E \mathcal{L}(E_i^g),\, \Delta E_i^g \right\rangle \\ &= \frac{1}{n_t}\sum_{i=1}^{n_t}\sum_{g=0}^{G_{\text{CA}}-1}\sum_{j=1}^{m_t}\frac{\partial \mathcal{L}(E_i^g)}{\partial E_{ij}^g}\,\Delta E_{ij}^g, \end{aligned} \tag{9}$$

where the Taylor remainder is bounded below under a Lipschitz-gradient assumption. The importance of concept j is

$$A_j^t \;=\; \frac{1}{n_t}\sum_{i=1}^{n_t}\sum_{g=0}^{G_{\mathrm{CA}}-1}\frac{\partial\mathcal{L}(E_i^g)}{\partial E_{ij}^g}\,\Delta E_{ij}^g, \qquad j=1,\dots,m_t. \tag{10}$$

This quantity is a *discrete path-integral* approximation along the exploration trajectory: a sum of inner products between the gradient (sensitivity) and the perturbation step, aligning with the integral-attribution form of (Sundararajan et al., 2017) (in discretized form). By construction via (8), the estimation leverages *all* samples in $\hat{\mathcal{D}}_t$.

For the default update, define $\bar{g}_j^g = n_t^{-1}\sum_i \partial\mathcal{L}(E_i^g)/\partial E_{ij}^g$. Substituting its shared increment into Eq. (10) gives the exact simplification

$$A_j^t = \eta_{\mathrm{CA}}\sum_{g=0}^{G_{\mathrm{CA}}-1}|\bar{g}_j^g| \geq 0. \tag{11}$$

CA therefore measures accumulated task-average gradient magnitude, depending on data, labels, loss, and path; weight magnitude depends on classifier parameters. Neither distinction implies predictive superiority. For a linear head this path probes a common class-logit offset; aggregate class-probability residuals can cancel completely even for an essential coordinate, as the example below shows. A zero score therefore does not establish unimportance. The bound below controls the total loss-decomposition error; Appendix A states the coordinate and path conditions for invariance.

Finite-step accuracy of the path decomposition. Let $N = n_t$, $G = G_{\mathrm{CA}}$, $m = m_t$, and $\eta = \eta_{\mathrm{CA}}$. The following statement characterizes the approximation in Eq. (9); its proof appears in Appendix A.

Proposition 3.1 (Default CA and finite-step remainder)**.** *Suppose $\eta \geq 0$ and Eq.* (8) *uses the same increment for every sample, with* $\mathrm{sign}(0) = 0$*. Then $A_j^t = \eta\sum_g |\bar{g}_j^g| \geq 0$. If each sample loss has an L-Lipschitz gradient on every segment $[E_i^g, E_i^{g+1}]$, then*

$$\left|\frac{1}{N}\sum_i\big[\mathcal{L}(E_i^G)-\mathcal{L}(E_i^0)\big]-\sum_j A_j^t\right| \leq \frac{L}{2N}\sum_{i,g}\|\Delta E_i^g\|_2^2 \leq \frac{LGm\eta^2}{2}. \tag{12}$$

For the specified affine frozen head with cross-entropy on logits, the concept Hessian is $W_J(\mathrm{diag}(p)-pp^\top)W_J^\top$, where W_J contains the current-block rows. Since the softmax covariance has spectral norm at most one, $L = \|W_J\|_2^2$ is a valid conservative global bound in that model. Differentiability alone would not justify a quadratic remainder; other heads or losses need their own smoothness bound.

What is decomposed. Equation (12) bounds the discrepancy between summed CA and loss change along the exploration path. Exploration is not concept deletion or retraining, so this is not an accuracy-loss or forgetting bound. At a fixed total coordinate travel budget $G\eta = B$, the displayed worst-case error is at most $LmB\eta/2$. Without such control, taking more steps alone does not guarantee a smaller error.

Shared-shift geometry and an exact blind spot. Because each sample receives the same increment, $E_i^g = E_i^0 + \delta^g$. For a fixed affine head and current-block rows W_J, the logits and mean gradient satisfy

$$z_i^g = z_i^0 + \delta^g W_J, \qquad \bar{g}^g = (\bar{p}^g - \bar{y})W_J^\top, \quad \bar{p}^g = N^{-1}\sum_i p_i^g, \quad \bar{y} = N^{-1}\sum_i \mathrm{onehot}(y_i).$$

Thus CA probes shared-shift loss sensitivity: it projects an aggregate class-probability residual onto the concept directions. If $\bar{p}^0 = \bar{y}$, then $\bar{g}^0 = 0$; with $\mathrm{sign}(0) = 0$, induction gives $\delta^g = 0$ and $A_j^t = 0$ for every step and coordinate. An unregularized class-bias optimum, when such biases are fitted, satisfies this mean-residual condition.

For an exact counterexample, take two equally weighted samples with concepts $E_1 = a, E_2 = -a$, labels 1 and 2, and logits $(wE, -wE)$, where $a, w > 0$. Both classify correctly. Their probabilities

are swapped and sum to $(1, 1)$, so their concept gradients cancel and the only coordinate has CA zero for every step. Deleting that coordinate gives logits $(0, 0)$ for both samples and 50% balanced accuracy under a deterministic lowest-class-index prediction tie rule. This is an analytical example, not a benchmark measurement. It demonstrates why the path loss bound does not imply deletion fidelity. How often such cancellation occurs in trained models remains empirical; replacing the shared shift with samplewise perturbations would change the analyzed method.

3.4 Concept Removal and Selecting the Pruning Ratio

We rank current-task concepts by $I_j^t = |A_j^t|$ and prune from low to high. For Eq. (8), $A_j^t \geq 0$, so the absolute value is redundant; this default score does not distinguish positive from negative contributions. Historical coordinates are retained.

Fixed-budget comparisons hold the count constant to isolate the ranking rule. The optional CV module compares candidate subsets k using the reported criterion

$$\mathrm{CV}_k = \frac{n^{-1} \sum_i \ell_i(W^k)}{1 - \gamma_k / n}, \qquad \gamma_k \approx \mathrm{tr}(H_k), \quad n = |\hat{\mathcal{D}}_t|. \tag{13}$$

Here W^k is the candidate classifier and $H_k \in \mathbb{R}^{nC_t \times nC_t}$ is a ridge smoothing matrix motivated by a local softmax quadratic approximation. Among admissible candidates, CV chooses the lowest score. The denominator requires $\gamma_k < n$, which ridge stabilization alone does not ensure. Equation (13) is a heuristic, not an exact or unbiased LOOCV identity; Appendix B gives the correct local algebra and its limits. The adaptive diagnostics require verification of their implementation and denominator handling before they can establish budget-selection effectiveness.

What the complexity correction approximates. For a fixed local ridge model, exact sample deletion acts on a coupled class-residual block as $e_i^{(-i)} = (I_{C_t} - H_{ii})^{-1} e_i$, not scalar division by $1 - \mathrm{tr}(H_{ii})$. Moreover, $\mathrm{tr}(H) < nC_t$ need not imply $\mathrm{tr}(H) < n$. Appendix B gives the spectral calculation and distinguishes scalar from block normalization; these facts do not derive the unsquared cross-entropy correction in Eq. (13).

4 Experiments

Evidence identity. The supplementary source uses per-sample gradient signs, rather than the shared mean-gradient sign in Eq. (8), and selects training states using a task test loader. Accordingly, all historical CA-labeled measurements below are *legacy-pipeline evidence*; they do not validate the analyzed CA rule or a strictly untouched test protocol. Appendix C records the source-level differences. Values are preserved without pooling later implementation versions.

4.1 Experimental Setup

Task protocol. We adopt the **class-incremental learning** protocol, where new classes arrive sequentially across tasks and **no exemplars are stored** (exemplar-free) (Yu et al., 2025). We denote splits as "B-m Inc-n": the first task contains m classes and each subsequent incremental task introduces n new classes until all classes are covered. The archived description specifies shared schedules and seeds, subject to the source-identity discrepancies above; archived whole-run costs are reported in Appendix D.1; isolated CA/CV overhead is unavailable. For full details of the training configuration and compute settings, please see Appendix D.

Datasets and splits. We evaluate on two **coarse-grained** and three **fine-grained** benchmarks: Tiny-ImageNet (Le et al., 2015) (200 classes: **B-10 Inc-10** and **B-100 Inc-10**); ImageNet-subset (Deng et al., 2009) (ImageNet-100: **B-10 Inc-10** and **B-50 Inc-5**); CUB-200 (Wah et al., 2011) (**B-10 Inc-10** and **B-100 Inc-10**); Stanford-Cars (Krause et al., 2013) (196 classes: **B-14 Inc-14**); Food-101 (Bossard et al., 2014) (101 classes, archive label **B-10 Inc-10**; remainder-task handling unresolved). All datasets use their standard train/val splits and conventional data augmentation.

Table 1: Results on **coarse-grained** benchmarks in the class-incremental setting (exemplar-free). Legacy LACE uses fixed 10% pruning. Values are Top-1 accuracy (%). **Bold** = best; underline = second best.

Methods	Tiny-ImageNet (B-10 Inc-10)		Tiny-ImageNet (B-100 Inc-10)		ImageNet-subset (B-10 Inc-10)		ImageNet-subset (B-50 Inc-5)	
	$\bar{A}$	A_{last}	$\bar{A}$	A_{last}	$\bar{A}$	A_{last}	$\bar{A}$	A_{last}
L2P (Wang et al., 2022b)	69.66	60.36	62.29	55.13	71.28	52.24	65.89	49.08
DualPrompt (Wang et al., 2022a)	74.06	66.08	65.35	56.15	72.86	54.20	64.64	49.48
CODA-Prompt (Smith et al., 2023)	75.18	66.65	61.47	48.31	71.17	52.98	64.64	41.40
CPP (Li et al., 2024)	68.70	61.23	66.61	63.48	83.45	75.80	79.74	73.78
LAE (Gao et al., 2023)	76.81	68.98	63.38	49.52	78.29	62.94	64.25	47.90
Continual-CLIP (Thengane et al., 2022)	68.66	55.91	58.68	55.91	84.98	75.40	81.35	75.40
SLCA (Zhang et al., 2023)	63.89	54.50	60.34	48.53	83.19	69.44	80.81	73.78
EASE (Zhou et al., 2024)	**79.88**	**72.99**	70.42	64.12	84.80	70.82	63.74	56.48
CLG-CBM (Yu et al., 2025)	79.28	71.98	75.64	71.97	86.83	78.97	81.85	78.21
LACE (Ours)	78.96	72.73	**76.13**	**72.43**	**93.35**	**89.72**	**91.69**	**89.94**

Baselines. We compare against representative methods: **L2P** (Wang et al., 2022b), **DualPrompt** (Wang et al., 2022a), **CODA-Prompt** (Smith et al., 2023), **CPP** (Li et al., 2024), **LAE** (Gao et al., 2023), **Continual-CLIP** (Thengane et al., 2022), **SLCA** (Zhang et al., 2023), **EASE** (Zhou et al., 2024), and **CLG-CBM** (Yu et al., 2025). The tables retain the originally reported comparisons; run provenance is required to certify matching schedules and test independence, including for **CLG-CBM**.

Metrics. Following class-incremental protocol, we report average accuracy $\bar{A}$ and final-stage accuracy A_{last}:

$$\bar{A} = \frac{1}{T}\sum_{t=1}^{T} A_t, \qquad A_{\text{last}} = A_T,$$

where A_t is the Top-1 accuracy on the *cumulative label space* after completing task t. These two metrics only indirectly reflect forgetting and concept control, so we avoid treating the $\bar{A}$-to-A_{last} gap as a substitute for BWT, per-task forgetting curves, or concept-count statistics. Table 1 presents $\bar{A}$ and A_{last} on the **coarse-grained** benchmarks (Tiny-ImageNet and ImageNet-subset), and Table 2 reports the corresponding results on the **fine-grained** benchmarks (CUB-200, Stanford-Cars, and Food-101).

Implementation details. Unless specified otherwise, the backbone is **CLIP ViT-B/16** (Radford et al., 2021). The manuscript reports pruning-rate bounds of 10%–40%; the archived CV row with 1,000 coordinates is inconsistent with these bounds and the stated cap (Appendix E.1). We use Adam with an initial learning rate of 1×10^{-3}, batch size 64, and train for 60 epochs per task with multi-stage scheduling and early stopping. These are the reported settings; Appendix C records conflicts with the supplied configuration. The concept space is instantiated from language and multimodal priors with a per-task *candidate cap of* k=100. The primary protocol is described as fixed 10% pruning after CA ranking so that accuracy and compactness are comparable across variants; CV-based budget selection is reported as a diagnostic in Appendix E.1. The supplied legacy scorer specifies ten steps of size 0.001; assigning exact configurations to reported runs remains unresolved. To curb forgetting under the exemplar-free setting, we employ prototype augmentation to synthesize lightweight pseudo-features that are trained jointly with the current task.

4.2 Results and Analysis

The archived table ordering is descriptive: unresolved protocol comparability and test selection prevent a verified superiority claim.

Table 1 reports the coarse-grained results. The archived LACE row is highest for Tiny-ImageNet B100/Inc10, whereas EASE is highest for the smaller-base B10/Inc10 split. This descriptive ordering does not establish the specified CA rule's effectiveness or a matched untouched-test comparison.

Fine-grained benchmarks. LACE also performs consistently on the fine-grained datasets. On CUB-200, it reaches average accuracies of 86.01% and 82.64% under the B-10 Inc-10 and B-100 Inc-10 settings. On Stanford-Cars, LACE achieves 89.56% average accuracy and 86.73% final-stage accuracy, improving over CLG-CBM by 0.96 and 1.66 percentage points. On Food-101, the

Table 2: Results on **fine-grained** benchmarks in the class-incremental setting (exemplar-free). For **CUB-200** we report $\bar{A}$ under two splits; for **Stanford-Cars** and **Food-101** we report both metrics. The primary protocol specifies fixed 10% pruning; † marks a value also reported under adaptive CA+CV in Appendix E.2, whose budget provenance requires run-log verification. Values are Top-1 accuracy (%). **Bold** = best; underline = second best.

	CUB-200		Stanford-Cars (B-14 Inc-14)		Food-101 (B-10 Inc-10)	
Methods	$\bar{A}$ (B-10 Inc-10)	$\bar{A}$ (B-100 Inc-10)	$\bar{A}$	A_{last}	$\bar{A}$	A_{last}
L2P (Wang et al., 2022b)	62.08	59.38	64.42	61.82	79.48	69.72
DualPrompt (Wang et al., 2022a)	64.95	61.85	76.94	68.46	86.27	69.66
CODA-Prompt (Smith et al., 2023)	67.22	59.82	76.44	60.80	87.76	67.22
CPP (Li et al., 2024)	83.60	75.03	84.75	77.49	90.21	86.60
LAE (Gao et al., 2023)	66.45	59.98	77.25	80.28	88.41	66.26
Continual-CLIP (Thengane et al., 2022)	69.41	60.35	86.43	69.79	92.04	89.96
SLCA (Zhang et al., 2023)	80.53	76.85	84.74	70.59	74.49	76.28
EASE (Zhou et al., 2024)	83.87	66.14	86.22	64.32	91.74	81.72
CLG-CBM (Yu et al., 2025)	<u>85.40</u>	<u>82.20</u>	<u>88.60</u>	<u>85.07</u>	<u>92.25</u>	<u>90.97</u>
LACE (Ours)	**86.01**	**82.64**†	**89.56**	**86.73**	**93.07**	**91.16**

corresponding results are 93.07% and 91.16%. The gains on Stanford-Cars are larger than those on CUB-200 and Food-101, suggesting that the benefit of concept-bank editing depends on the visual granularity and task sequence of the benchmark.

On **ImageNet-subset (IN-100)**, the gains are substantially larger. With **B-10 Inc-10**, we reach $\bar{A} = \mathbf{93.35}$ and $A_{\text{last}} = \mathbf{89.72}$, exceeding **CLG-CBM** (86.83/78.97) by $+6.52/+10.75$ pp; with **B-50 Inc-5**, we obtain $\mathbf{91.69}/\mathbf{89.94}$, improving over **CLG-CBM** by $+9.84/+11.73$ pp. These end-to-end margins motivate the component comparison below; they do not isolate the contribution of CA.

Table 3 makes the archived IN-100 component evidence visible. Prototype augmentation has the largest accuracy effect. At the same 900-coordinate budget, CA and magnitude are close; removing pruning retains 100 additional coordinates and slightly raises accuracy. These legacy summaries describe a count–accuracy trade-off; they do not validate the specified CA ordering or an untouched-test comparison.

Table 3: Archived legacy IN-100 B-10 Inc-10 ablations; mean ± SD over seeds 1993–1995. Ratios are fixed; full diagnostics are in Appendix E.1.

Variant	$\bar{A}$ (%)	A_{last} (%)	Concepts
Legacy CA, fixed 10%	93.35 ± 0.46	89.72 ± 0.20	900
No prototype augmentation	28.92 ± 0.02	9.87 ± 0.08	900
No pruning	93.78 ± 0.35	89.97 ± 0.22	1000
Magnitude, fixed 10%	93.32 ± 0.51	89.60 ± 0.29	900

On the fine-grained benchmarks (Table 2), LACE has the largest displayed values, but the CUB B100 cell's budget provenance remains unresolved. Food-101's final-stage difference is only $+0.19$ pp; accuracy gaps do not measure forgetting. Appendix E gives the detailed comparisons and diagnostics.

5 Conclusion

LACE separates task-local ranking, coordinate budgets, and prototype-based training support in continual CBMs. The finite-step analysis characterizes the specified CA rule; archived ablations expose the accuracy–cardinality trade-off of the legacy pipeline and the large prototype-augmentation effect. Source differences and test-based selection prevent transferring these measurements to the analyzed rule or an untouched-test evaluation. The shared-shift counterexample identifies a concrete ranking failure mode, while the block-deletion calculation specifies what a trace correction omits. Resolving implementation identity remains necessary, alongside validating adaptive budgets and semantic usefulness.

AI USE STATEMENT

Generative AI tools assisted with manuscript refinement, language clarity and readability, and LaTeX formatting. The authors are responsible for the final manuscript.

REFERENCES

Sylvain Arlot and Alain Celisse. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4:40–79, 2010. doi: 10.1214/09-SS054.

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.

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

Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. *Advances in neural information processing systems*, 32, 2019.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Qiankun Gao, Chen Zhao, Yifan Sun, Teng Xi, Gang Zhang, Bernard Ghanem, and Jian Zhang. A unified continual learning framework with general parameter-efficient tuning. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11449–11459. IEEE, 2023.

Gene H Golub, Michael Heath, and Grace Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 24(12):1028–1040, 2020.

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pp. 2668–2677. PMLR, 2018.

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.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pp. 5338–5348. Pmlr, 2020.

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.

Yann Le, Xuan Yang, et al. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

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

Zhuowei Li, Long Zhao, Zizhao Zhang, Han Zhang, Di Liu, Ting Liu, and Dimitris N Metaxas. Steering prototypes with prompt-tuning for rehearsal-free continual learning. In *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 2511–2521. IEEE, 2024.

Meike Nauta, Ron Van Bree, and Christin Seifert. Neural prototype trees for interpretable fine-grained image recognition. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14928–14938. IEEE, 2021.

Tuomas Oikarinen, Subhro Das, Lam M Nguyen, and Tsui-Wei Weng. Label-free concept bottleneck models. *arXiv preprint arXiv:2304.06129*, 2023.

Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*, 2018.

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.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 5533–5542. IEEE, 2017.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Dawid Rymarczyk, Joost Van De Weijer, Bartosz Zieliński, and Bartlomiej Twardowski. Icicle: Interpretable class incremental continual learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1887–1898, 2023.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

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 *2023 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11909–11919. IEEE, 2023.

Mervyn Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the royal statistical society: Series B (Methodological)*, 36(2):111–133, 1974.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pp. 3319–3328. PMLR, 2017.

Vishal Thengane, Salman Khan, Munawar Hayat, and Fahad Khan. Clip model is an efficient continual learner. *arXiv preprint arXiv:2210.03114*, 2022.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

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, 2022a.

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 *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 139–149. IEEE, 2022b.

Weibin Wu, Yuxin Su, Xixian Chen, Shenglin Zhao, Irwin King, Michael R Lyu, and Yu-Wing Tai. Towards global explanations of convolutional neural networks with concept attribution. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8649–8658. IEEE, 2020.

Yue Yang, Artemis Panagopoulou, Shenghao Zhou, Daniel Jin, Chris Callison-Burch, and Mark Yatskar. Language in a bottle: Language model guided concept bottlenecks for interpretable image classification. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19187–19197. IEEE, 2023.

Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in neural information processing systems*, 33:20554–20565, 2020.

Lu Yu, Haoyu Han, Zhe Tao, Hantao Yao, and Changsheng Xu. Language guided concept bottleneck models for interpretable continual learning. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14976–14986. IEEE, 2025.

Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. *arXiv preprint arXiv:2205.15480*, 2022.

Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. Slca: Slow learner with classifier alignment for continual learning on a pre-trained model. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19091–19101. IEEE, 2023.

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

Zhiyu Zhu, Huaming Chen, Jiayu Zhang, Xinyi Wang, Zhibo Jin, Minhui Xue, Dongxiao Zhu, and Kim-Kwang Raymond Choo. Mfaba: A more faithful and accelerated boundary-based attribution method for deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17228–17236, 2024.

A Finite-Step Attribution Analysis

The analysis concerns the specified mean-gradient sign update with frozen model parameters and a fixed set of concept coordinates. Let $N = n_t$, $G = G_{\mathrm{CA}}$, $m = m_t$, and $\eta = \eta_{\mathrm{CA}}$.

Proof. Substitution of $\Delta E_{ij}^g = \eta \operatorname{sign}(\bar{g}_j^g)$ in Eq. (10) gives $\eta \sum_g \bar{g}_j^g \operatorname{sign}(\bar{g}_j^g)$. For each step, the integral remainder is

$$R_i^g = \int_0^1 \langle \nabla \mathcal{L}(E_i^g + u\Delta E_i^g) - \nabla \mathcal{L}(E_i^g), \Delta E_i^g \rangle \, du.$$

The Lipschitz assumption gives $|R_i^g| \le L\|\Delta E_i^g\|_2^2/2$. Sum the Taylor identities over steps, telescope, average over samples, and use the triangle inequality. Finally, $|\Delta E_{ij}^g| \le \eta$ yields the second bound. □

Cancellation and invariance. Gradients $+1$ and -1 from two samples cancel in the task average, even when both sample losses depend on the coordinate. CA therefore does not satisfy general pointwise Sensitivity. If two differentiable implementations represent the same loss function on the same concept coordinates in a neighborhood of all sampled points, and use identical initialization, data, and deterministic update rules, their gradients and hence their CA paths coincide in exact arithmetic. This follows by induction over steps. It does not imply invariance under a change of concept basis, stochastic execution, or nondifferentiable tie conventions. Optional sample-dependent clipping would invalidate the nonnegativity simplification in general and must be specified as a different path rule.

B Local Quadratic Reference for the CV Heuristic

This appendix gives a mathematically consistent local reference for a complexity correction. It does not identify the executed adaptive implementation from the manuscript alone. In particular, the reported scalar criterion in Eq. (13) is retained as a heuristic; it does not follow as an exact LOOCV formula from the algebra below.

Fixed local model and dimensions. For one candidate subset with q retained coordinates and all $C_t = |\mathcal{Y}_{1:t}|$ seen classes (write $C = C_t$ locally in this appendix), fix its design $X \in \mathbb{R}^{n\times q}$, a linearization point W_0, probabilities p_i, regularization constants $\epsilon, \rho > 0$, and the resulting working responses and weights. All are held fixed during the following deletion calculation. Let $x_i \in \mathbb{R}^q$ be a column, $\theta = \mathrm{vec}(W) \in \mathbb{R}^{qC}$ use column-major vectorization, and

$$D_i = I_C \otimes x_i^\top \in \mathbb{R}^{C\times qC}, \qquad \eta_i = D_i\theta_0, \quad S_i = \operatorname{diag}(p_i) - p_i p_i^\top, \quad S_i^\epsilon = S_i + \epsilon I_C.$$

Let $u_i = \mathrm{onehot}(y_i) \in \mathbb{R}^C$. The softmax loss has gradient $p_i - u_i$ and logit Hessian S_i. Adding ϵI_C stabilizes a local quadratic, rather than leaving that Hessian unchanged. It also adds curvature in the common-logit direction, where true softmax probabilities are invariant, so the ridge trace can count directions that do not change predictions. Centering logits or using a reference class would change this heuristic and requires separate validation. Completing its square gives

$$z_i = \eta_i + (S_i^\epsilon)^{-1}(u_i - p_i), \qquad Q_i(D_i\theta) = \tfrac{1}{2}\|(S_i^\epsilon)^{1/2}(D_i\theta - z_i)\|_2^2 + \mathrm{const}.$$

The weight is the *positive* square root of the covariance; replacing it with the inverse square root changes the objective. Stack blocks in sample-major order,

$$\widetilde{X} = \begin{bmatrix} (S_1^\epsilon)^{1/2} D_1 \\ \vdots \\ (S_n^\epsilon)^{1/2} D_n \end{bmatrix} \in \mathbb{R}^{nC\times qC}, \qquad \widetilde{z} = \begin{bmatrix} (S_1^\epsilon)^{1/2} z_1 \\ \vdots \\ (S_n^\epsilon)^{1/2} z_n \end{bmatrix}.$$

Explicit D_i blocks avoid mixing sample-major covariance blocks with a class-major stacked design. For the ridge quadratic

$$\min_\theta \ \tfrac{1}{2}\|\widetilde{z} - \widetilde{X}\theta\|_2^2 + \tfrac{\rho}{2}\|\theta\|_2^2,$$

the fitted vector and smoothing matrix are

$$\hat{\widetilde{z}} = H\widetilde{z}, \qquad H = \widetilde{X}(\widetilde{X}^\top \widetilde{X} + \rho I)^{-1}\widetilde{X}^\top \in \mathbb{R}^{nC\times nC}. \tag{14}$$

This ridge reference is distinct from fitting the full nonlinear softmax objective or a nonsmooth sparse penalty.

Deleting one sample block. Write $e = \widetilde{z} - H\widetilde{z}$ and let $H_{ii} \in \mathbb{R}^{C\times C}$ be the diagonal sample block. Removing the entire sample while holding working quantities fixed gives the exact quadratic residual identity

$$e_i^{(-i)} = (I_C - H_{ii})^{-1} e_i. \tag{15}$$

To see this, put $A = \widetilde{X}^\top \widetilde{X} + \rho I$, and let B_i be the sample block of $\widetilde{X}$. The deleted normal equations imply $(A - B_i^\top B_i)(\theta^{(-i)} - \hat{\theta}) = -B_i^\top e_i$. Equivalently $A(\theta^{(-i)} - \hat{\theta}) = -B_i^\top e_i^{(-i)}$; substitution in $e_i^{(-i)} = e_i - B_i(\theta^{(-i)} - \hat{\theta})$ proves the identity. Ridge positivity makes $A - B_i^\top B_i$ positive definite. Scalar division by $1 - \mathrm{tr}(H_{ii})$ is generally incorrect for this coupled class block.

What the complexity correction approximates. The local ridge reference also makes the distinction from sample deletion explicit. For a fixed candidate with q retained coordinates and $C = C_t$ classes, stack the positively weighted, sample-major design as $\widetilde{X} \in \mathbb{R}^{nC\times qC}$ and its working response as $\widetilde{z} \in \mathbb{R}^{nC}$. The weights and responses are fixed at a softmax linearization point, as specified in Appendix B. For ridge coefficient $\rho > 0$,

$$H = \widetilde{X}(\widetilde{X}^\top \widetilde{X} + \rho I)^{-1}\widetilde{X}^\top, \qquad e = \widetilde{z} - H\widetilde{z}.$$

If $H_{ii} \in \mathbb{R}^{C\times C}$ is the diagonal block for sample i, its exact deleted-sample residual in this fixed quadratic problem is

$$e_i^{(-i)} = (I_C - H_{ii})^{-1} e_i.$$

Deleting one sample therefore involves an inverse acting on its coupled class residual. Replacing this inverse by division by $1 - \mathrm{tr}(H_{ii})$ is generally incorrect. Ridge positivity ensures that the deleted normal equations are invertible; it does not make the class coordinates independent.

If s_a are the nonzero singular values of $\widetilde{X}$, then

$$\mathrm{tr}(H) = \sum_a \frac{s_a^2}{s_a^2 + \rho} < nC.$$

This upper bound is in scalar working-response coordinates and need not be smaller than n. The average scalar leverage is $\mathrm{tr}(H)/(nC)$, whereas $\mathrm{tr}(H)/n$ averages the block traces. Even an isotropic block approximation would use the former for its diagonal entries. Finally, squared-residual GCV squares its denominator (Golub et al., 1979); it does not derive Eq. (13), which uses held-in cross-entropy and an unsquared correction. The reported CV rule thus combines several approximations that cannot be removed merely by calling the trace an effective dimension. A candidate set must specify its weighting, normalization, and admissibility handling before its adaptive budget is reproducible.

Scope of the reported score. Using Eq. (13) involves a local quadratic model, a trace summary replacing block leverage, and a heuristic correction of held-in cross-entropy. It neither recomputes representations after deletion nor measures exact leave-one-out loss. Candidate pruning subsets and prototype pseudo-features are themselves data-dependent, so the fixed-design argument also does not establish unbiased generalization estimation. The score is meaningful as a positive correction only when $\gamma_k < n$, preferably bounded away from n. The run configuration must record the actual weighting, stacking, normalization, and behavior when that condition fails. Correcting those choices would change the adaptive procedure and requires revalidation; no such change or new measurement is claimed here.

C Source Identity and Archived Evidence

Read-only inspection of the supplied supplementary `code/models/lace.py` identifies the following distinctions. They are reported explicitly to separate the mathematical reference from historical execution; no training or scoring implementation was changed for this revision.

- `prune_model` iterates over training batches and updates each sample by its own gradient sign for ten steps of size 0.001, accumulating gradient times displacement and dividing by ten before averaging samples. Equation (8) instead takes the sign after averaging gradients over the task. Even in exact arithmetic, mean absolute gradient and absolute mean gradient differ; a common positive normalization does not remove this distinction. The legacy code also uses automatic mixed precision, and its actual displacement can differ from the intended small update.
- `main_pruning` uses a fixed `self.ratio`; its declared candidate-ratio array is unused in the inspected file. Thus this archived source does not establish execution of the hat-matrix criterion. Separate historical audit notes additionally report that a later adaptive path evaluated candidates on the passed task test loader; that version is not pooled with these tables.
- `incremental_train` constructs the training-selection loader from `source='test'`. Both training and post-pruning finetuning evaluate this loader when selecting states. The displayed historical accuracies therefore cannot be presented as an audited untouched-test evaluation. The exact checkpoint and configuration that generated each table cell must be identified before a clean protocol claim can be made.
- The supplied CUB B100/Inc10 configuration specifies 60 initial and 20 incremental head-training epochs, while the prose configuration below states a common 60-epoch schedule. This discrepancy also requires run-level provenance rather than silently choosing one description.

Unresolved reproducibility fields. The available record does not identify a consistent run-level LLM/version and prompt/filtering policy, cap handling, feature normalization, full sparsity coefficients, trainable historical blocks, candidate-classifier fit schedule, or CV ϵ, ρ and inadmissibility fallback. Food-101's B10/Inc10 label also leaves unresolved whether a final one-class task was used or only 100 classes were evaluated. These require original run records; they are not guessed here. Source constants for the legacy scorer do not certify settings of the analyzed reference or every displayed table row.

An archived July 2026 source audit records formula mismatch and mixed-precision degradation, including constant-zero legacy scores at several inspected tasks. This revision independently confirms the samplewise-versus-mean update difference in the supplementary source, but does not rerun that numerical audit or promote its measurements into accuracy evidence. Theoretical results here apply to the displayed reference update only.

D Training Configuration and Compute Budget

The following is the archived manuscript configuration, retained for traceability. It is not a verified run-level specification: Appendix C records a conflicting CUB schedule and selection-loader behavior. LACE-specific profiling is reported in Appendix D.1.

- **Backbone:** CLIP ViT-B/16 (frozen image encoder).
- **Training epochs per task:** 60.
- **Batch size:** 64.
- **Optimizer:** Adam.
- **Learning rate:** 0.001.
- **Random seed:** 1993 by default; multi-seed analyses use seeds 1993, 1994, and 1995 when explicitly reported.
- **CA exploration:** the supplied legacy scorer uses ten steps of size 0.001; exact run mapping is unverified.
- **Data usage:** No external data; each task uses only its designated subset.

The archive reports whole-run costs for two configurations, without isolating CA or CV. The legacy source includes post-pruning classifier finetuning; this stage is not part of the frozen-parameter CA

analysis, and its exact run-level schedule is unresolved. Candidate fitting for the optional CV heuristic is likewise unspecified beyond the fixed local quadratic reference. Frozen CLIP features do not establish matched head-training work or equal compute across methods.

D.1 Additional Results on Training Cost and Memory Overhead

In this section, we provide empirical measurements for the *lightweight* claim. The table below reports the currently available LACE measurements on the CUB-200 dataset with ViT-B/16 under two commonly used class-incremental settings. They are absolute reported costs, with unresolved timing and memory boundaries; no isolated overhead or matched efficiency conclusion follows from them.

Table 4: Archived training-cost and reported extra-memory figures; measurement boundaries are unresolved.

Dataset	Model	Setting	Pruning rate	Time per last task (s)	Total time (s)	Total time (hours)	Extra memory
CUB-200	ViT-B/16	B10 Inc10	0.1	55	12660	3h 31m	≈ 1 MB ± 0.1
CUB-200	ViT-B/16	B100 Inc10	0.1	56	5340	1h 29m	≈ 0.9 MB ± 0.1

These archived fixed-rate measurements describe the reported LACE run, rather than isolated CA or CV costs. They do not establish the overhead of adaptive candidate fitting or the peak memory of a dense hat matrix. Exact timing boundaries, devices, and buffer accounting are needed to reconcile per-task and whole-sequence costs and to reproduce the measurements. For example, 1,800 retained mapping rows of dimension 512 alone require about 3.69 MB in FP32, conditional on the stated full-cap CUB B10 protocol. Reconciling the reported approximately 1 MB extra-memory value requires its dtype, count, and exclusion conventions; the table cannot establish an independent lightweight-overhead claim.

E Additional Ablations and Qualitative Analysis

On the **fine-grained** benchmarks (Table 2), the archived LACE values are largest among the displayed methods: CUB-200 differs from **CLG-CBM** by $+0.61/+0.44$ pp in the two splits (the B100 cell is excluded from fixed-budget conclusions because its provenance is unresolved), Stanford-Cars by $+0.96/+1.66$ pp in $\bar{A}/A_{\text{last}}$, and Food-101 by $+0.82/+0.19$ pp. The Food-101 final-stage gain is small, and the $\bar{A}$-to-A_{last} gap alone does not measure forgetting; we therefore treat these results as accuracy evidence and defer direct compactness, forgetting, and qualitative analyses to Appendix E.

E.1 IN-100 Component Ablations

Table 5 reports the completed 3-seed component ablation on ImageNet-subset (IN-100) under the B-10 Inc-10 split. The largest effect comes from prototype augmentation: removing it reduces final accuracy to 9.87%, a severe decline but above the 1% uniform-chance rate for 100 classes. The scalar alone does not identify which old or new tasks fail. The no-pruning variant slightly improves accuracy while retaining more concepts, so we interpret the fixed-budget pruning result as evidence for a compactness–accuracy trade-off rather than as an accuracy-only improvement.

Table 5: IN-100 B-10 Inc-10 core component ablations. Values are averaged over seeds 1993, 1994, and 1995; accuracy columns report mean $\pm$ standard deviation.

Variant	$\bar{A}\uparrow$	$A_{\text{last}}\uparrow$	Concepts $\downarrow$
LACE (legacy CA, fixed 10% pruning)	93.35 ± 0.46	89.72 ± 0.20	900
No prototype augmentation	28.92 ± 0.02	9.87 ± 0.08	900
No pruning budget	93.78 ± 0.35	89.97 ± 0.22	1000
Weight-magnitude pruning	93.32 ± 0.51	89.60 ± 0.29	900

Table 6: IN-100 B-10 Inc-10 single-seed diagnostic ablations. All runs use seed 1993. The adaptive CV row is an archived diagnostic whose executed criterion requires verification (Appendix B); it does not validate the corrected local reference.

Variant	$\bar{A}\uparrow$	$A_{\text{last}}\uparrow$	Concepts $\downarrow$
Random pruning	92.60	89.42	900
Gradient-absolute pruning	92.62	89.32	900
Taylor pruning	92.62	89.32	900
Activation-mean pruning	92.63	89.62	900
CV (budget provenance unresolved)	93.16	89.78	1000
Linear activation	92.69	89.74	900
Gaussian-only prototype	60.02	39.04	900
Relation-only prototype	91.27	88.40	900
Random-matched prototype	92.70	89.30	900

Forgetting metrics. For task-accuracy matrix $R_{t,i}$ in percentage points, standard final average forgetting and backward transfer are

$$F_T = \frac{1}{T-1}\sum_{i<T}\big[\max_{i\le t\le T} R_{t,i} - R_{T,i}\big], \qquad \mathrm{BWT}_T = \frac{1}{T-1}\sum_{i<T}(R_{T,i} - R_{i,i}).$$

The archived ablation column labeled "Forgetting" lacks a recoverable definition and unit in the available records; it is withheld rather than reinterpreted as either statistic. Task-accuracy matrices are required to validate these readouts. No new forgetting measurements are asserted here.

Table 6 provides additional single-seed diagnostics. These runs are useful for understanding design sensitivity, but we do not use them as primary evidence for component superiority. In particular, several simple pruning scores are close to CA on this split, and the archived CV-budget diagnostic selects a larger concept bank in the reported seed, subject to the implementation checks in Appendix B. These observations motivate a conservative claim: CA supplies an auditable concept-ranking mechanism, while compact pruning can reduce the concept bank with little accuracy loss; stronger statements about CA dominating all alternative scores require broader multi-seed evidence.

With ten tasks, at most 100 candidates per task, and at least 10% removed from every current block, at most 900 coordinates remain. The archived CV count of 1,000 therefore cannot follow all three specifications. Its actual candidate set, cap, or counting convention is unresolved; it is not evidence for compactness within the stated bounds. The "Gaussian-only," "Relation-only," and "Random-matched" prototype labels are also retained as archive labels only: operational definitions and matching configs are required before these rows identify component effects.

E.2 Ablation of Concept Attribution (CA) and Concept Verification (CV)

The archive labels the following comparison as an ablation of Concept Attribution (CA) and Concept Verification (CV) on **CUB-200** with **ViT-B/16** under the **B100 Inc10** setting. We consider three variants:

- **CA only**: LACE with CA but without CV (we use a fixed pruning ratio instead of data-driven verification).
- **CV only**: LACE with CV but replacing CA with a naive magnitude-based ranking of concepts.
- **CA + CV (full LACE)**: our full method using both CA and CV.

The average accuracy $\bar{A}$ over all tasks is reported in Table 7.

The recorded full-model accuracy is higher in this comparison. This does not isolate the effect of ranking at a matched retained count, and the available table does not report uncertainty. Because the executed adaptive implementation needs the checks in Appendix B, these values are preserved as archived diagnostics rather than evidence for exact LOOCV or optimal adaptive pruning.

Table 7: Archived CA/CV comparison on CUB-200 with ViT-B/16 (B100 Inc10). CA-only uses a fixed budget; CV-only and CA+CV use adaptive budgets. The adaptive implementation is not certified by Appendix B.

Dataset	Model	Setting	Components	$\bar{A}$
CUB-200	ViT-B/16	B100 Inc10	CA only	81.53
CUB-200	ViT-B/16	B100 Inc10	CV only	81.61
CUB-200	ViT-B/16	B100 Inc10	CA + CV (full LACE)	**82.64**

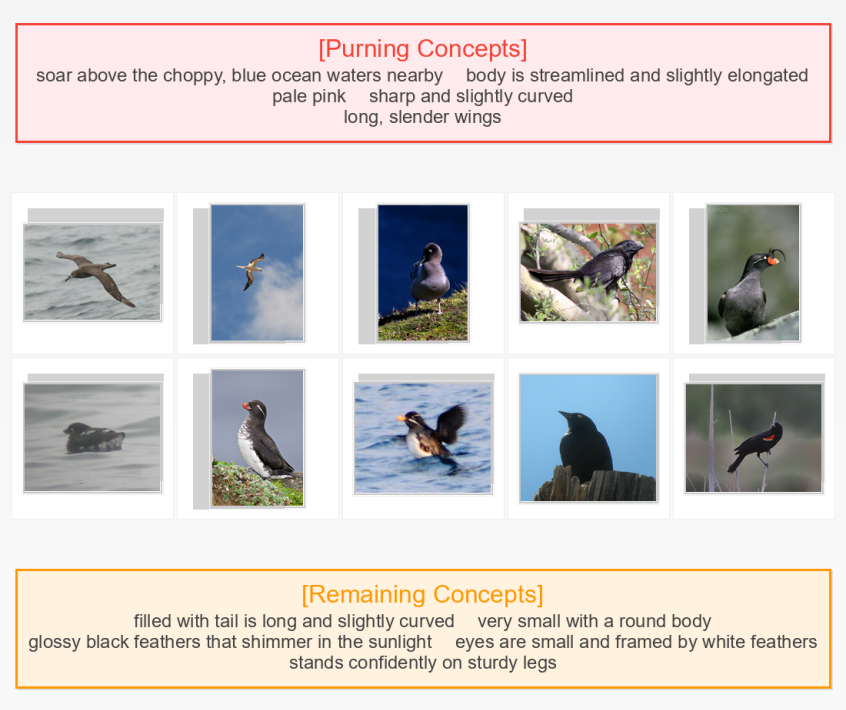


Figure 2: Qualitative visualization on CUB-200 with ViT-B/16. Top: text labeled pruned. Middle: ten bird images. Bottom: text labeled retained. Image-to-concept correspondence and class membership are unspecified; this panel does not establish semantic superiority or human interpretability.

E.3 Qualitative Analysis of Pruned vs. Retained Concepts

Figure 2 is an archived illustrative panel labeled CUB-200, with pruned text at the top, a middle grid of ten bird images, and retained text at the bottom. Class labels, sample IDs, score values, and image-to-concept correspondences are unspecified. Selection follows the CA ordering and chosen budget; CV sets a count rather than identifying individual concepts. The panel includes background descriptions among the removed coordinates and appearance descriptions among those retained. It also includes "sharp and slightly curved long, slender wings" among the removed concepts, showing that the removed text can name a meaningful attribute; the panel does not provide its score or verify a pairing to particular images. It does not establish population-level grounding, cross-task stability, or human usability.

F Notation Table

Table 8 summarizes the key mathematical symbols and notations used in the description of the LACE framework.

Table 8: Summary of Mathematical Notations.

Symbol	Description
	Problem Definition & Data
$\mathcal{T}$	Task index set $\{1, 2, \ldots, T\}$.
$\mathcal{D}_t$	Dataset available for the current task t, consisting of pairs (x_i, y_i).
$\hat{\mathcal{D}}_t$	Feature-level training set for task t, including current CLIP features and augmented pseudo-features.
$\mathcal{Y}_t, \mathcal{Y}_{1:t}$	Label set for task t and cumulative label space up to task t.
$E_I(\cdot), E_T(\cdot)$	CLIP image and text encoders, mapping to $\mathbb{R}^d$ ($d = 512$).
	CBM Architecture & Prototype Augmentation
$\mathcal{P}_t^{\text{new}}, \hat{\mathcal{P}}_{1:t-1}, \hat{\mathcal{P}}_{1:t}^{\text{post}}$	New task concepts, retained historical concepts, and retained post-pruning pool.
$M_t, \hat{M}_t$	Number of generated concepts and number of retained concepts for task t.
$\mathbf{C}_t$	Bank of concept vectors encoded by $E_T(\cdot)$ for task t.
$\mathbf{W}_c^t$	Learnable parameters of the Concept Bottleneck Layer (CBL) at task t.
$\mathbf{W}_l^t$	Learnable parameters of the linear classification head at task t.
$\mathcal{L}_{\text{sim}}$	Similarity loss aligning concept scores with CLIP's concept space.
P_i, P_j^{old}	Current class prototype and cached old-class prototype in CLIP image-feature space.
$\tilde{\mathbf{V}}_{\text{pseudo}}^j$	Set of pseudo-feature vectors generated for old class j using semantic matching.
	Concept Attribution (CA)
E_i^g	Concept representation of sample i at exploration step g.
ΔE_i^g	Increment of concept representation between steps g and $g + 1$.
$\eta_{\text{CA}}, G_{\text{CA}}$	Step size and number of exploration steps for CA in concept space.
A_j^t	Importance score of concept j at task t (discrete path-integral approximation).
	Concept Verification (CV)
CV_k	Heuristic complexity-corrected loss for candidate pruning ratio k.
H_k	Hat matrix used to approximate the effective complexity of the model.
γ_k	Effective complexity term, approximated by $\text{tr}(H_k)$.
$\tilde{X}_k$	Scaled design matrix used in the IRLS approximation.
S_i^ϵ	Regularized softmax covariance matrix for sample i, $S_i^\epsilon \in \mathbb{R}^{C_t \times C_t}$.