LEARNING TO COMPOSE PROMPTS WITH HEBBIAN PATHWAYS FOR CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Parameter-efficient continual learning adapts a frozen pre-trained model with small modules, such as prompts, that hold the knowledge acquired over the task sequence. However, allocating modules to each task avoids interference but prevents reuse and enlarges the pool with every task, whereas sharing modules across tasks enables reuse but exposes the shared modules to interference from later tasks. This paper proposes a new method of Learning to Compose (L2C) that represents the task knowledge as a composition of prompts selected at multiple layers of a vision transformer, thereby lowering the unit of sharing and protecting from tasks to prompts. Under this representation, a new task can be expressed as a new composition that reuses the prompts of earlier tasks. L2C determines which prompts to protect from a selection history that is decoupled from the task loss, in which a Hebbian rule accumulates the selection probabilities of prompts chosen together at adjacent layers. The history serves as a prior that stabilizes compositions already in use and determines which prompts are frozen or re-initialized orthogonally to the frozen keys, recovering capacity without enlarging the pool. Experiments on three benchmarks with 5 to 40 tasks show that L2C attains the highest average incremental accuracy and the lowest forgetting against ten baselines in all nine settings, while training fewer than 0.2% of the backbone parameters.

1 INTRODUCTION

Continual learning with pre-trained models has largely converged on parameter-efficient continual learning (PEFT-CL), in which the backbone is frozen and the knowledge acquired over a task sequence is held in a small set of modules, such as prompts (Wang et al., 2022b;a; Smith et al., 2023) or adapters (Zhou et al., 2024b; Sun et al., 2025; Wang et al., 2025), which are selected or combined for each input without task identity at test time (Zhou et al., 2024a; Wang et al., 2024). Existing methods organize these modules across tasks in one of two ways. Allocation methods assign new modules to each task and keep them fixed once the task has been learned (Wang et al., 2022a; Zhou et al., 2024b; Sun et al., 2025; Wang et al., 2025), whereas sharing methods maintain a single set of modules that every task uses and updates (Wang et al., 2022b; Gao et al., 2023).

Neither organization scales with the length of the task sequence: allocation accumulates modules and candidates for selection, whereas sharing accumulates interference. Under allocation, freezing a module protects what it contains but not whether it is selected, and as later tasks add candidates, the prompts selected for an input of an earlier task depart from those used to train it; the task-specific prompts of DualPrompt are selected with less than 50% accuracy on ImageNet-R by the tenth task (Gao et al., 2023), and MOS refers to the resulting mismatch as retrieval-level forgetting (Sun et al., 2025). Under sharing, the pool stays fixed, but its prompts and keys are updated on the data of each new task through the task loss or a surrogate matching loss, so the selection on which earlier tasks rely drifts (Menabue et al., 2024); the same effect has been reported for learned attention over parameter-efficient modules in continual learning of language models (Zhao et al., 2024). Subsequent methods reduce these costs within each organization, for example by aligning the selection between training and testing (Gao et al., 2024) or by adding a universal module shared by all tasks to the task-specific ones (Wang et al., 2025).

However, both organizations decide what to share and what to protect for a task as a whole rather than for individual modules. Allocation keeps the knowledge of each task apart from the other tasks,

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.

and sharing exposes all of it to later tasks, so neither can reuse part of what a task has learned while protecting the part on which earlier tasks rely. The backbone offers a finer unit: the early blocks of a pre-trained vision transformer (ViT) attend more locally and resemble the lower layers of convolutional networks, whereas the late blocks represent more global and semantic information (Raghu et al., 2021; Ghiasi et al., 2022), so tasks with different high-level semantics may still share low-level processing. Deciding sharing and protection at this unit further requires a record of which modules earlier tasks rely on, which a selection fitted to the data of the current task does not retain.

This paper proposes a method of Learning to Compose (L2C) that represents the knowledge of a task as a composition of prompts selected at multiple layers of a ViT, thereby lowering the unit of sharing and protection from tasks to prompts. The prompts of each layer are selected with a query computed at the same depth of the frozen, prompt-free backbone, so a new task can be expressed as a new composition that reuses the prompts of earlier tasks at the layers where it shares processing with them and selects other prompts from which it differs, within a pool of fixed size. The remaining question, which prompts to protect, is answered by a selection history that receives no gradient from the task loss: Oja's Hebbian rule (Oja, 1982) accumulates the selection probabilities of prompts chosen together at adjacent layers into transition matrices, which weight the pathways between prompts across layers. The history enters the selection score as a prior conditioned on the selection at the preceding layer, which stabilizes the compositions in use. It also decides which prompts are frozen and which are re-initialized: the most used prompts are frozen, and their number grows only logarithmically with the number of tasks, whereas the least used ones receive new keys orthogonal to the frozen keys, which recovers capacity for new compositions without enlarging the pool. The contributions are as follows.

- A composition of layer-wise prompts for PEFT-CL that decides sharing and protection for each prompt rather than for each task, from a selection history accumulated by a Hebbian rule without gradients from the task loss (Section 3).
- An analysis of why this history protects earlier tasks at a fixed pool size: it records the use of each prompt and changes only the connections of prompts in use (Proposition 1 and Corollary 1), its prior widens the selection margin of the compositions in use (Proposition 2), and re-initialized keys are orthogonal to the frozen keys and nearly orthogonal to earlier queries with high probability (Proposition 3).
- Experiments on Split CIFAR-100, ImageNet-R, and ImageNet-A with 5 to 40 tasks, in which L2C attains the highest average incremental accuracy and the lowest forgetting against ten baselines in all nine settings, and removing the shared shallow prompts raises forgetting the most (Section 4).

2 Related Work

Prompt-based continual learning. L2P selects prompts from a shared pool by matching a query from the frozen backbone against learnable keys trained with a surrogate loss (Wang et al., 2022b). DualPrompt attaches a shared G-prompt and task-specific E-prompts to fixed layers and selects the E-prompt through task keys (Wang et al., 2022a), and CODA-Prompt composes the prompt as an input-conditioned weighted sum of components, adding components for each task and learning the weights end-to-end with the task loss (Smith et al., 2023). HiDe-Prompt optimizes within-task prediction, task-identity inference, and task-adaptive prediction separately with stored representation statistics (Wang et al., 2023a). Subsequent work targets the selection itself: CPrompt aligns the training of prompts and classifiers with test-time selection (Gao et al., 2024), VQ-Prompt makes the discrete selection differentiable so that it is optimized with the task loss (Jiao et al., 2024), and STAR-Prompt selects prompts with a two-level mechanism built on CLIP whose class prototypes are frozen after their task (Menabue et al., 2024); PGP projects prompt gradients to reduce interference (Qiao et al., 2024), and Le et al. (2024) interpret prefix prompts as mixtures of experts. Most of these methods select or compose prompts once per input from a query at a single depth. L2C composes prompts selected at multiple depths and decides which of them to protect from a selection history that receives no gradient from the task loss.

Adapter, low-rank, and expansion methods. RanPAC projects frozen features through a random nonlinear layer and fits a closed-form classifier (McDonnell et al., 2023), and APER adapts the backbone on the first task and merges the adapted and pre-trained features (Zhou et al., 2025);

SLCA (Zhang et al., 2023) and LAE (Gao et al., 2023) update shared parameters with a small learning rate or an online-offline ensemble. Expansion methods allocate modules per task: EASE adds an adapter subspace per task and concatenates their features (Zhou et al., 2024b), MOS merges task-specific adapters and retrieves one with a self-refined procedure (Sun et al., 2025), and TUNA combines a task-specific adapter selected by entropy with a universal adapter (Wang et al., 2025). InfLoRA and SD-LoRA constrain or decouple low-rank updates to limit interference (Liang & Li, 2024; Wu et al., 2025), and MoE-Adapters route among adapter experts in a vision-language model (Yu et al., 2024). MEMO shares the shallow blocks of the network and expands its deep blocks for each task (Zhou et al., 2023), which fixes by depth what is shared. L2C instead decides sharing and protection for individual prompts rather than by depth, keeps a pool of fixed size, and exposes only the prompts it has not frozen to later tasks.

Module selection in continual learning of language models. O-LoRA learns each task in a low-rank subspace orthogonal to those of earlier tasks (Wang et al., 2023b), and Progressive Prompts learns a new soft prompt per task and concatenates it with the earlier ones (Razdaibiedina et al., 2023). SAPT combines per-task modules through instance-level attention and restores the attention of earlier tasks with generated pseudo-samples (Zhao et al., 2024), and SLM retrieves stored weight increments with keys that are learned in a separate stage and then frozen (Peng et al., 2024). For multimodal models, the CoIN benchmark evaluates a mixture of low-rank experts whose gate is trained with the task loss (Chen et al., 2024), and HiDe-LLaVA keeps task-specific modules only in the top layer, selected by similarity to stored anchors, after finding that lower layers hold more generic knowledge (Guo et al., 2025), which likewise fixes the depth at which knowledge is shared. These remedies (freezing or separately training the selector, matching on fixed features, and replaying the selection) mirror those in vision; the experiments of this paper are limited to vision.

Compositional structure and Hebbian learning. Compositional continual learning reuses modules across tasks and learns task-specific combinations of them (Mendez & EATON, 2021; Ostapenko et al., 2021), and re-initializing a small fraction of less-used units maintains plasticity in deep continual learning (Dohare et al., 2024). Hebbian plasticity (Hebb, 2005) has been used to train deep networks without feedback (Journé et al., 2022) and to extract principal subspaces of neural activity for orthogonal projection in continual learning of spiking networks (Xiao et al., 2024); both apply the rule to feature weights. L2C uses Oja's rule (Oja, 1982) to the selection rather than to the features: the rule accumulates which prompts are selected together at adjacent depths, and the resulting history both stabilizes the compositions in use and decides which prompts are frozen or re-initialized.

3 Proposed Method

3.1 Problem Formulation

Tasks arrive in a sequence $t = 1, \ldots, T$ with disjoint label sets $\mathcal{Y}_t$, and the data $\mathcal{D}_t$ of task t are available only while it is being learned; at test time, an input is classified among all classes seen so far without task identity, and no exemplars are stored. A frozen backbone with L blocks (ViT-B/16, $L = 12$) is adapted by a pool of prompts, of which a set $S(\boldsymbol{x})$ is selected for each input $\boldsymbol{x}$. For an input of class j in task k,

$$P\big(y = (k, j) \mid \boldsymbol{x}\big) = P(j \mid \boldsymbol{x}, k)\, P(k \mid \boldsymbol{x}), \qquad H_{\mathrm{CIL}}(\boldsymbol{x}) = H_{\mathrm{WP}}(\boldsymbol{x}) + H_{\mathrm{TP}}(\boldsymbol{x}), \tag{1}$$

where $H_{\mathrm{WP}} = -\log P(j \mid \boldsymbol{x}, k)$ measures within-task prediction and $H_{\mathrm{TP}} = -\log P(k \mid \boldsymbol{x})$ task-identity prediction (Kim et al., 2022). The prediction for $\boldsymbol{x}$ depends on both $S(\boldsymbol{x})$ and the contents of the selected prompts, so freezing prompts protects what they contain but leaves the task-identity factor exposed whenever later tasks change $S(\boldsymbol{x})$. The goal is to learn the sequence with a pool of fixed size such that (R1) knowledge is shared and protected per prompt rather than per task, so that a new task can reuse part of what earlier tasks learned; (R2) the record that decides which prompts to protect is not trained by the task loss, and the queries with which prompts are selected do not change after the first task; and (R3) capacity is recovered for new tasks without enlarging the pool and without biasing earlier selections toward the recovered prompts.

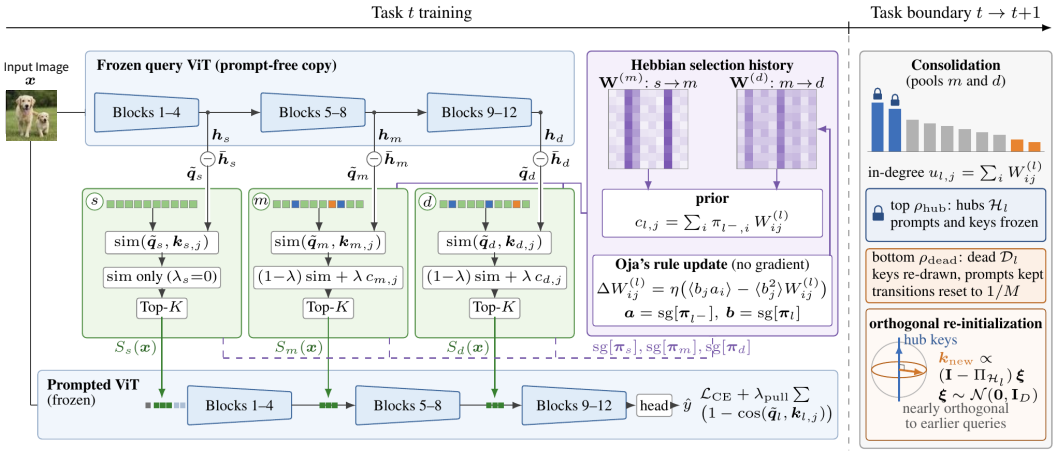


Figure 1: Overview of L2C. (Left) Prompts are selected at three stages of ViT-B/16 (blocks 1–4, 5–8, and 9–12) with queries from a frozen copy of the backbone. (Middle) The selection score combines query-key similarity with a prior from a selection history between adjacent stages, which a Hebbian rule accumulates from detached selection probabilities. (Right) At task boundaries, prompts with a large history are frozen as hubs, and the keys of prompts with a small history are re-initialized in the orthogonal complement of the hub keys.

3.2 Stage-wise Prompt Pools and Centered Queries

L2C selects prompts at several depths of the backbone (Fig. 1), since the early blocks of a pre-trained ViT attend more locally and resemble lower convolutional layers whereas the late blocks are more global (Raghu et al., 2021; Ghiasi et al., 2022). The 12 blocks of ViT-B/16 are divided into three stages $l \in \{s, m, d\}$ of four blocks each (1–4, 5–8, and 9–12): the within-stage similarity of the frozen backbone, measured by centered kernel alignment (Kornblith et al., 2019), increases most when the number of contiguous stages grows from two to three, and the boundaries are uniform rather than tuned (Section B). Each stage has a pool of M prompts $\boldsymbol{P}_{l,j} \in \mathbb{R}^{L_p \times D}$ with keys $\boldsymbol{k}_{l,j} \in \mathbb{R}^D$, $j = 1, \ldots, M$. The K prompts selected at a stage (Section 3.3) are inserted after the `[CLS]` token at the first block of the stage (blocks 1, 5, and 9) and remain in the token sequence for the subsequent blocks, so a new task can share the prompts of some stages with earlier tasks while selecting different prompts at the others (R1). The prompts thus act as the atoms of a dictionary shared by all tasks and the selection sets as sparse codes, as in multi-task sparse coding (Section A.4).

The query of stage l is derived from $\boldsymbol{h}_l(\boldsymbol{x})$, the `[CLS]` output of the last block of the stage (blocks 4, 8, and 12, before the final layer normalization) in a frozen, prompt-free copy of the backbone. These outputs share a large component common to all images: without centering, the similarity to the keys varies across inputs by only 5% of its variation across prompts, and on 20-task ImageNet-R every input selects the same prompts at each stage (Section D.5). L2C therefore centers the queries,

$$\tilde{\boldsymbol{q}}_l(\boldsymbol{x}) = \boldsymbol{h}_l(\boldsymbol{x}) - \bar{\boldsymbol{h}}_l, \tag{2}$$

where $\bar{\boldsymbol{h}}_l$ is the mean of $\boldsymbol{h}_l$ over the data of the first task. During the first task, $\bar{\boldsymbol{h}}_l$ is the running mean of the queries seen so far, including the current mini-batch; it is fixed when the first task ends, so the query of every input stays unchanged over the rest of the sequence (R2).

3.3 Selection with a Hebbian History

Selecting each stage independently ignores which prompts have been selected together and how often. L2C records this history in non-negative transition matrices $\boldsymbol{W}^{(m)}, \boldsymbol{W}^{(d)} \in \mathbb{R}^{M \times M}$, where $\boldsymbol{W}^{(l)}$ weights the pathways from the prompts of the preceding stage l^- to those of stage l and is initialized to $1/M$. The score of prompt j at stage l is

$$s_{l,j}(\boldsymbol{x}) = (1 - \lambda_l)\,\mathrm{sim}\big(\tilde{\boldsymbol{q}}_l(\boldsymbol{x}), \boldsymbol{k}_{l,j}\big) + \lambda_l\, c_{l,j}(\boldsymbol{x}), \qquad c_{l,j}(\boldsymbol{x}) = \sum_{i=1}^{M} \pi_{l^-,i}(\boldsymbol{x})\, W_{ij}^{(l)}, \tag{3}$$

where $\mathrm{sim}(\boldsymbol{q}, \boldsymbol{k}) = \frac{1}{2}\big(1 + \cos(\boldsymbol{q}, \boldsymbol{k})\big) \in [0, 1]$, $\boldsymbol{\pi}_l(\boldsymbol{x}) = \mathrm{softmax}(\boldsymbol{s}_l(\boldsymbol{x})/\kappa)$ is the selection distribution, $\lambda_s = 0$, and $\lambda_m = \lambda_d = \lambda$. The K prompts with the highest scores form $S_l(\boldsymbol{x})$. After each parameter update, the matrices are updated by Oja's rule (Oja, 1982) on the detached selection distributions of consecutive stages,

$$\Delta W_{ij}^{(l)} = \eta\big(\langle b_j a_i\rangle - \langle b_j^2\rangle W_{ij}^{(l)}\big), \qquad \boldsymbol{a} = \mathrm{sg}\left[\boldsymbol{\pi}_{l^-}(\boldsymbol{x})\right], \;\; \boldsymbol{b} = \mathrm{sg}\left[\boldsymbol{\pi}_l(\boldsymbol{x})\right], \tag{4}$$

where $\langle\cdot\rangle$ denotes the average over the mini-batch. The matrices receive no gradient, and the task loss reaches them only through the selection distributions (R2). Proposition 1 states what they store, with Δ^{M-1} the probability simplex and $u_j = \sum_i W_{ij}$ the in-degree of prompt j; all proofs are given in Section A.

Proposition 1 (Hebbian transition matrix)**.** *Consider one transition updated only by Eq.* (4)*, from $W_{ij}(0) = 1/M$ and with $\eta \in (0, 1]$.* (a) *$W_{ij}(n) \geq 0$ and $u_j(n) \geq 1$ for all n.* (b) *If the mini-batches are drawn i.i.d. within a task, the selection distributions are fixed functions of the input, and $\mathbb{E}[b_j^2] > 0$, then $\mathbb{E}[W_{ij}(n)] - W_{ij}^\star = (1 - \eta\mathbb{E}[b_j^2])^n\,(W_{ij}(0) - W_{ij}^\star)$ with $W_{ij}^\star = \mathbb{E}[a_i b_j]/\mathbb{E}[b_j^2]$, and the fixed-point column is $\boldsymbol{W}_{\cdot j}^\star = \rho_j \boldsymbol{\mu}_j$ with $\boldsymbol{\mu}_j = \mathbb{E}[b_j \boldsymbol{a}]/\mathbb{E}[b_j] \in \Delta^{M-1}$ and $\rho_j = \mathbb{E}[b_j]/\mathbb{E}[b_j^2] \geq 1$.* (c) *For $n_1 < n_2$, $\big\|\boldsymbol{W}_{\cdot j}(n_2) - \boldsymbol{W}_{\cdot j}(n_1)\big\|_1 \leq \eta \sum_{n=n_1}^{n_2-1} \big(\langle b_j\rangle_n + \langle b_j^2\rangle_n\, u_j(n)\big)$, where $\langle\cdot\rangle_n$ is the average over the mini-batch of step n.*

The matrices serve as a selection history that spans the task sequence rather than a statistic of a single task. By (b), column j of $\boldsymbol{W}$ moves toward a fixed point given by a scaled average of the earlier-stage selection distributions of the inputs that select prompt j, weighted by how strongly they select it, and the deviation of $\mathbb{E}[\boldsymbol{W}_{\cdot j}]$ from it shrinks by the factor $1 - \eta\,\mathbb{E}[b_j^2]$ per update. On 20-task ImageNet-R, the selection probabilities stay below 0.13, so $\eta\,\mathbb{E}[b_j^2] < 2 \times 10^{-4}$ for $\eta = 0.01$, and the matrices accumulate the use of each prompt over the whole sequence instead of converging to the fixed point of the current task, which reflects that task alone. They operate in the regime of Corollary 1, in which the expected in-degree grows by about $\eta\,\mathbb{E}[b_j(1 - b_j)]$ per update, a quantity that increases with the selection probability while that probability stays below $1/2$; accordingly, the in-degree follows the mean selection probability, with Spearman correlations of 0.84 and 0.85 in the middle and deep pools (Section D.5). The in-degree thus records how much each prompt has been used, and Section 3.4 relies on it to decide which prompts to protect. By (c), the change of a column is bounded by the selection probabilities of its prompt, so later tasks rewrite the history of a prompt only as much as they select it. The decay term distinguishes Oja's rule from a co-occurrence count, which grows without bound; Section 4.3 compares the two.

The prior carries this history into the selection: $c_{l,j}(\boldsymbol{x})$ follows the in-degree of prompt j and is stable across inputs, whereas the similarity of the centered query varies strongly across inputs (Section D.5). The two terms of Eq. (3) thus play complementary roles: the similarity adapts the selection to the current input, and the prior ranks the prompts by their past use, consistently for every input. For one stage and input, let $\boldsymbol{v}, \boldsymbol{c} \in \mathbb{R}^M$ collect the similarities and the priors, let $\boldsymbol{s} = (1 - \lambda)\boldsymbol{v} + \lambda\boldsymbol{c}$ and $S = \mathrm{TopK}(\boldsymbol{s})$, and let $g(\boldsymbol{z}) = \min_{j \in S,\, j' \notin S}(z_j - z_{j'})$ denote the margin of S under scores $\boldsymbol{z}$.

Proposition 2 (Margin of the selection)**.** (a) *For perturbations $\boldsymbol{e}_v, \boldsymbol{e}_c \in \mathbb{R}^M$ of the similarities and the prior,* $\mathrm{TopK}\big((1-\lambda)(\boldsymbol{v} + \boldsymbol{e}_v) + \lambda(\boldsymbol{c} + \boldsymbol{e}_c)\big) = S$ *whenever $(1-\lambda)\|\boldsymbol{e}_v\|_\infty + \lambda\|\boldsymbol{e}_c\|_\infty < g(\boldsymbol{s})/2$.* (b) *If $S = \mathrm{TopK}(\boldsymbol{v})$ and $\gamma = \min_{j \in S} c_j - \max_{j \notin S} c_j \geq 0$, then $g(\boldsymbol{s}) \geq (1 - \lambda)g(\boldsymbol{v}) + \lambda\gamma$. Hence, for $\boldsymbol{e}_c = \boldsymbol{0}$, the similarity perturbation tolerated by the score, $g(\boldsymbol{s})/(2(1 - \lambda))$, exceeds that of similarity-only selection, $g(\boldsymbol{v})/2$, by at least $\lambda\gamma/(2(1 - \lambda))$.* (c) *Since the queries are fixed after the first task, $\|\boldsymbol{e}_v\|_\infty \leq \frac{1}{2}\max_j \|\hat{\boldsymbol{k}}'_{l,j} - \hat{\boldsymbol{k}}_{l,j}\|_2$, where $\hat{\boldsymbol{k}} = \boldsymbol{k}/\|\boldsymbol{k}\|$ and $\boldsymbol{k}'$ denotes the updated key; $e_{v,j} = 0$ for frozen keys.*

Proposition 2 connects the components of L2C. Key updates enter the selection of an earlier input only through (c), and hub freezing removes them for hub keys; the prior widens the margin when the selected prompts are those used more in the past, as in (b); and by Proposition 1(c), the part of $\boldsymbol{e}_c$ caused by changes of $\boldsymbol{W}$ is bounded by the use of the candidate prompts in later tasks. For example, with $K = 1$, $\boldsymbol{v} = (0.62, 0.60)$, $\boldsymbol{c} = (0.8, 0.2)$, and $\lambda = 0.5$, a change of 0.02 in the similarity difference flips similarity-only selection, whereas the score of Eq. (3) flips only after a change of 0.62. A prior that disagrees with the similarities ($\gamma < 0$) can override them, which is why $\lambda < 1$ (Section 4.3).

3.4 Hub Freezing and Orthogonal Re-initialization

At the end of each task, L2C reallocates the middle and deep pools according to the history; the shallow pool has no incoming transitions and is neither frozen nor re-initialized. Among the prompts of stage l that are not yet hubs, the top ρ_{hub} fraction by in-degree $u_{l,j}$ is added to the hub set $\mathcal{H}_l$ as long as $|\mathcal{H}_l| \le \lceil \log_2(t+2) \rceil$ after task t (Section C), and the prompts and keys of hubs are frozen by masking their gradients. The bottom ρ_{dead} fraction of the remaining prompts forms the dead set $\mathcal{D}_l$, whose keys are re-drawn as

$$\boldsymbol{k}_{\mathrm{new}} = \frac{(\boldsymbol{I} - \Pi_{\mathcal{H}_l})\boldsymbol{\xi}}{\|(\boldsymbol{I} - \Pi_{\mathcal{H}_l})\boldsymbol{\xi}\|}, \qquad \boldsymbol{\xi} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_D), \tag{5}$$

where $\Pi_{\mathcal{H}_l} = \boldsymbol{Q}\boldsymbol{Q}^\top$ is the orthogonal projection onto the span of the hub keys, with $\boldsymbol{Q}$ an orthonormal basis of that span obtained by QR decomposition and the projection applied twice for numerical stability. The prompt tokens of a dead prompt are kept, and its transition weights are reset to $1/M$.

Proposition 3 (Re-initialized keys)**.** *Let $r < D$ be the dimension of the span of the hub keys, $m = D - r$, and $\sin\theta(\boldsymbol{q}) = \|(\boldsymbol{I} - \Pi_{\mathcal{H}_l})\boldsymbol{q}\|/\|\boldsymbol{q}\|$ for $\boldsymbol{q} \neq \boldsymbol{0}$.* (a) *$|\cos(\boldsymbol{q}, \boldsymbol{k}_{\mathrm{new}})| \le \sin\theta(\boldsymbol{q})$ for every $\boldsymbol{q}$; in particular, $\cos(\boldsymbol{q}, \boldsymbol{k}_{\mathrm{new}}) = 0$ for every $\boldsymbol{q}$ in the span of the hub keys.* (b) *$\boldsymbol{k}_{\mathrm{new}}$ is uniformly distributed on the unit sphere of the orthogonal complement of that span, and for any fixed $\boldsymbol{q}$ and $\epsilon \in (0,1)$, $P\big(|\cos(\boldsymbol{q}, \boldsymbol{k}_{\mathrm{new}})| \ge \epsilon \sin\theta(\boldsymbol{q})\big) \le 2e^{-m\epsilon^2/2}$.* (c) *For any finite set $\mathcal{Q}$ of queries, $P\big(\max_{\boldsymbol{q}\in\mathcal{Q}} |\cos(\boldsymbol{q}, \boldsymbol{k}_{\mathrm{new}})| \ge \epsilon\big) \le 2|\mathcal{Q}|e^{-m\epsilon^2/2}$.*

In the experiments, $D = 768$ and at most $\lceil \log_2 42 \rceil = 6$ hubs per stage give $m \ge 762$, so by (c) a re-initialized key has cosine similarity below 0.2 with all 2.4×10^4 centered training queries of ImageNet-R with probability at least 0.98. A re-initialized prompt therefore starts at the neutral similarity $\mathrm{sim} \approx 1/2$ for earlier inputs, exactly so for inputs whose centered queries lie in the span of the hub keys, and acquires its role through the alignment loss on new-task data (R3).

3.5 Objective and Procedure

Prompts, keys, and a linear classifier over the classes seen so far are trained with

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}}(\hat{y}, y) + \lambda_{\mathrm{pull}} \sum_{l\in\{s,m,d\}} \sum_{j\in S_l(\boldsymbol{x})} \big(1 - \cos(\tilde{\boldsymbol{q}}_l(\boldsymbol{x}), \boldsymbol{k}_{l,j})\big), \tag{6}$$

where the logits of the classes of earlier tasks are masked in $\mathcal{L}_{\mathrm{CE}}$, and the second term aligns the selected keys with the centered queries so that the similarity in Eq. (3) remains informative. Because the selection is a hard top-K, the keys receive gradients only from this alignment term. Within a task, each step updates the non-hub prompts and keys by the masked gradient of Eq. (6) and then the matrices by Eq. (4); at the end of the task, hubs are added and dead keys re-initialized (Algorithm 1 in Section C). The trainable parameters, counted as the prompts, the keys, and the classifier rows of the current task, amount to fewer than 0.2% of the backbone (Section D.10); apart from the classifier, they do not depend on T, and the update of $\boldsymbol{W}$ costs $O(M^2)$ per sample without backpropagation.

4 Experiments

4.1 Setup

Benchmarks and protocol. Split CIFAR-100 (Krizhevsky, 2009) is divided into 5, 10, and 20 tasks, and Split ImageNet-R (Hendrycks et al., 2021a) and ImageNet-A (Hendrycks et al., 2021b) into 10, 20, and 40 tasks (Section C); VTAB (Zhai et al., 2019), whose domains lie far from the pre-training data, is analyzed separately in Section D.1. All methods use ViT-B/16 (Dosovitskiy et al., 2021) pre-trained on ImageNet-21K and fine-tuned on ImageNet-1K, kept frozen, and are trained for 5 epochs per task with Adam (Kingma & Ba, 2014), learning rate 10^{-3}, and batch size 16, so that every method receives the same number of iterations per task, following the evaluation of continual learning under a computational budget (Prabhu et al., 2023; Seo et al., 2025); baselines follow the PILOT toolbox (SUN et al., 2023). Each of five seeds permutes the class order. Methods designed for longer schedules or growing architectures (HiDe-Prompt, MOS, TUNA) are compared at 5 and 20 epochs in Section D.2. L2C uses $M = 10$, $L_p = 5$, $K = 5$, $\lambda = 0.5$, $\kappa = 1$, $\eta = 0.01$, and $\lambda_{\mathrm{pull}} = 0.1$ throughout (Table 5); results with DeiT backbones are given in Section D.9.

Table 1: Average incremental accuracy (Avg, %), forgetting (Fgt, %), and final accuracy on the longest sequence of each benchmark (A_T, %), mean over five seeds; standard deviations are given in Tables 7 and 8. Best results are in **bold** and second-best results underlined.

	CIFAR-100						Final accuracy A_T		
	5 tasks		10 tasks		20 tasks		CIFAR-100	ImageNet-R	ImageNet-A
Method	Avg	Fgt	Avg	Fgt	Avg	Fgt	20 tasks	40 tasks	40 tasks
L2P	84.36	7.23	81.94	7.65	78.14	8.97	70.62	56.24	18.36
DualPrompt	87.84	8.54	86.45	9.12	82.54	11.76	75.31	62.86	43.77
CODA-P	86.81	10.01	85.50	10.92	81.97	12.05	73.51	59.12	44.15
RanPAC	90.19	3.66	88.58	4.41	87.72	4.98	80.54	66.63	47.88
EASE	90.18	7.04	88.53	7.67	87.10	7.53	80.54	64.47	52.38
LAE	88.49	7.32	86.81	7.96	83.25	9.89	76.00	60.90	43.34
MEMO	68.32	29.69	74.17	30.04	37.43	37.17	30.76	41.40	8.50
APER-A	89.31	5.24	88.14	6.00	86.69	6.58	79.09	64.51	55.54
APER-S	89.46	4.84	89.13	5.45	88.00	5.82	80.53	67.53	57.17
APER-V	89.53	4.90	88.51	5.61	86.25	6.00	78.57	64.53	56.56
L2C	**90.41**	**3.29**	**89.71**	**3.51**	**88.79**	**4.16**	**80.88**	**68.99**	**59.72**

	ImageNet-R						ImageNet-A					
	10 tasks		20 tasks		40 tasks		10 tasks		20 tasks		40 tasks	
Method	Avg	Fgt	Avg	Fgt	Avg	Fgt	Avg	Fgt	Avg	Fgt	Avg	Fgt
L2P	72.07	6.76	64.03	8.54	63.80	7.16	54.00	15.08	46.20	9.75	31.00	10.11
DualPrompt	74.11	7.73	71.07	10.33	69.67	8.34	62.34	9.42	60.16	9.99	54.26	12.12
CODA-P	74.41	8.09	73.12	9.13	68.53	10.79	61.86	10.74	61.67	10.40	55.00	11.51
RanPAC	78.12	5.94	75.06	6.22	72.24	6.39	65.94	9.38	63.47	9.67	60.76	22.45
EASE	70.80	6.98	70.69	7.18	70.92	7.47	66.00	11.64	64.82	12.24	62.19	12.44
LAE	74.01	6.90	71.54	8.43	68.67	9.38	62.25	10.20	59.81	12.30	55.13	12.44
MEMO	69.51	23.28	69.43	25.43	50.49	29.44	31.79	34.21	25.79	32.28	14.93	21.54
APER-A	70.46	6.10	70.63	6.43	70.94	6.71	65.05	10.21	64.85	10.88	62.40	11.28
APER-S	78.05	5.34	76.62	6.04	71.70	6.37	65.99	10.23	63.29	11.00	61.31	11.46
APER-V	76.39	5.13	74.39	5.61	71.05	6.02	60.13	10.13	63.79	10.52	61.10	11.01
L2C	**79.24**	**3.56**	**77.41**	**4.40**	**73.10**	**5.74**	**66.47**	**7.49**	**65.11**	**8.99**	**63.57**	**9.78**

Baselines and metrics. The baselines are L2P, DualPrompt, CODA-Prompt (CODA-P), RanPAC, EASE, LAE, MEMO, and APER with an adapter, SSF, or VPT (APER-A/S/V). With $a_{t,i}$ the accuracy on task i after learning task t and A_t the accuracy on all classes seen after task t (Section D.7), the average incremental accuracy is $\mathrm{Avg} = \frac{1}{T}\sum_{t=1}^{T} A_t$, and forgetting is $\mathrm{Fgt} = \frac{1}{T-1}\sum_{i<T} \max_{i\le t<T}(a_{t,i} - a_{T,i})$ (Chaudhry et al., 2018). Table 1 also reports the final accuracy A_T on the longest sequence of each benchmark.

4.2 Comparison with Existing Methods

L2C attains the highest Avg and the lowest Fgt in all nine settings and the highest final accuracy on the longest sequence of each benchmark (Table 1), which shows that deciding what to share and what to protect for individual prompts avoids the costs that the two organizations of parameters incur. The prompt-based methods share one query–key selection across all tasks, and the strongest of them trails L2C by 2.57 to 8.57 points in Avg; on CIFAR-100 and ImageNet-A, the gap is largest in the longest sequence (6.25 and 8.57 points), where the most tasks train the shared selection. The expansion methods, EASE and MEMO, allocate modules to each task and still forget more than L2C in every setting, although the parameters of L2C do not grow with the number of tasks. RanPAC and APER fix their representation after the first task, which limits forgetting by construction; L2C, which trains prompts for every task, forgets less than all of them in every setting while attaining higher Avg, so the selection history protects earlier tasks without giving up adaptation to later ones. The advantage holds under longer training as well: with 20 epochs per task on 10-task ImageNet-R,

Table 2: Ablations, mean over five seeds. (a) Components on 20-task ImageNet-R: hierarchical composition (HC), selection history (SH; Hebbian prior and hub freezing), orthogonal re-initialization (OR), and query centering (QC). (b) Update rule for $\boldsymbol{W}$ on 40-task ImageNet-R, as the difference from Oja's rule. (c) Stages composed on 40-task ImageNet-A.

(a)

Variant	HC	SH	OR	QC	Avg	Fgt
Single pool	✗	✗	✗	✗	65.94	10.31
+ HC	✓	✗	✗	✗	71.33	8.29
+ SH	✓	✓	✗	✗	73.12	5.26
+ OR	✓	✓	✓	✗	75.68	5.32
L2C	✓	✓	✓	✓	**77.41**	**4.40**

(b)

Rule	ΔAvg	ΔFgt
Count	-5.24	$+2.91$
EMA	-2.86	$+1.28$
Oja	0	0

(c)

Stages	Avg	Fgt
s, m	54.92	12.40
m, d	57.00	14.33
s, d	60.52	13.14
s, m, d	**63.57**	**9.78**
4 stages	58.69	12.83

a larger configuration of L2C with six pools of 20 prompts exceeds HiDe-Prompt, MOS, and TUNA by 0.90 to 12.59 points in Avg and forgets 2.23 to 10.07 points less (Section D.2).

4.3 Ablation Study

Each component of L2C fulfills the role for which it is designed (Table 2(a)). Composing prompts across the hierarchy raises Avg from 65.94 to 71.33 and lowers Fgt from 10.31 to 8.29, as the combinations add capacity for new tasks while the prompts shared at the lower levels stabilize earlier ones (Table 2(c)). The selection history lowers Fgt from 8.29 to 5.26, and from 12.78 to 9.04 on 20-task ImageNet-A (Section D.3), which confirms that protecting the most used prompts preserves earlier tasks. Orthogonal re-initialization raises Avg from 73.12 to 75.68 while Fgt stays within 0.06 points, as Proposition 3 anticipates for keys that start orthogonal to the hub keys, and query centering, which makes the selection input-dependent, raises Avg to 77.41 and lowers Fgt to 4.40. The selection history also guides the selection itself (Table 12): without the prior in the score ($\lambda = 0$), Avg drops to 75.48 although hubs and re-initialization still use the history, and the prior alone ($\lambda = 1$) reaches 76.36, above the similarity alone; their combination performs best, in line with Proposition 2. The alignment term serves plasticity: removing it lowers Avg to 75.50 while Fgt remains 4.24, and weights from 0.1 to 1.0 keep Avg within 0.29 points (Section D.4).

The selection history relies on the use-proportional decay of Oja's rule (Table 2(b)). Replacing it by raw co-occurrence counts lowers Avg by 5.24 points and raises Fgt by 2.91 points, and replacing it by an exponential moving average lowers Avg by 2.86 points and raises Fgt by 1.28 points. Counts grow without bound, so the prior outweighs the similarity in Eq. (3) and fixes the selection regardless of the input; the moving average stays bounded but decays every entry at the same rate, which erases the history of prompts that later tasks rarely select. Oja's rule decays each column only in proportion to the use of its target prompt (Proposition 1(c)), so the matrices stay bounded while the history of prompts that later tasks rarely select is preserved.

Composing prompts from all three stages performs best (Table 2(c)). On 40-task ImageNet-A, removing any stage lowers Avg by 3.05 to 8.65 points and raises Fgt by 2.62 to 4.55 points. The two ends of the hierarchy play complementary roles, consistent with the selection diagnostics on ImageNet-R, where the inputs of a task form a median of 10 distinct selection sets in the shallow stage and 32 in the deep stage (Section D.5): removing the widely shared shallow stage raises Fgt the most (14.33), and removing the input-specific deep stage lowers Avg the most (54.92). A fourth stage lowers Avg to 58.69, in agreement with the analysis of the backbone by centered kernel alignment (Section B).

4.4 Selection History and Efficiency

L2C composes prompts for individual inputs, and its selection history records which of these compositions are in use. On 20-task ImageNet-R, every middle and deep prompt is selected, and the inputs of a task form a median of 10, 24, and 32 distinct selection sets in the shallow, middle, and deep stages (Section D.5). From the uniform initialization, the mean row entropy of the transition matrices decreases and their Gini coefficient increases as tasks accumulate (Fig. 2, top), and the two matrices develop different structures (Fig. 2, bottom): in the shallow-to-middle matrix, many

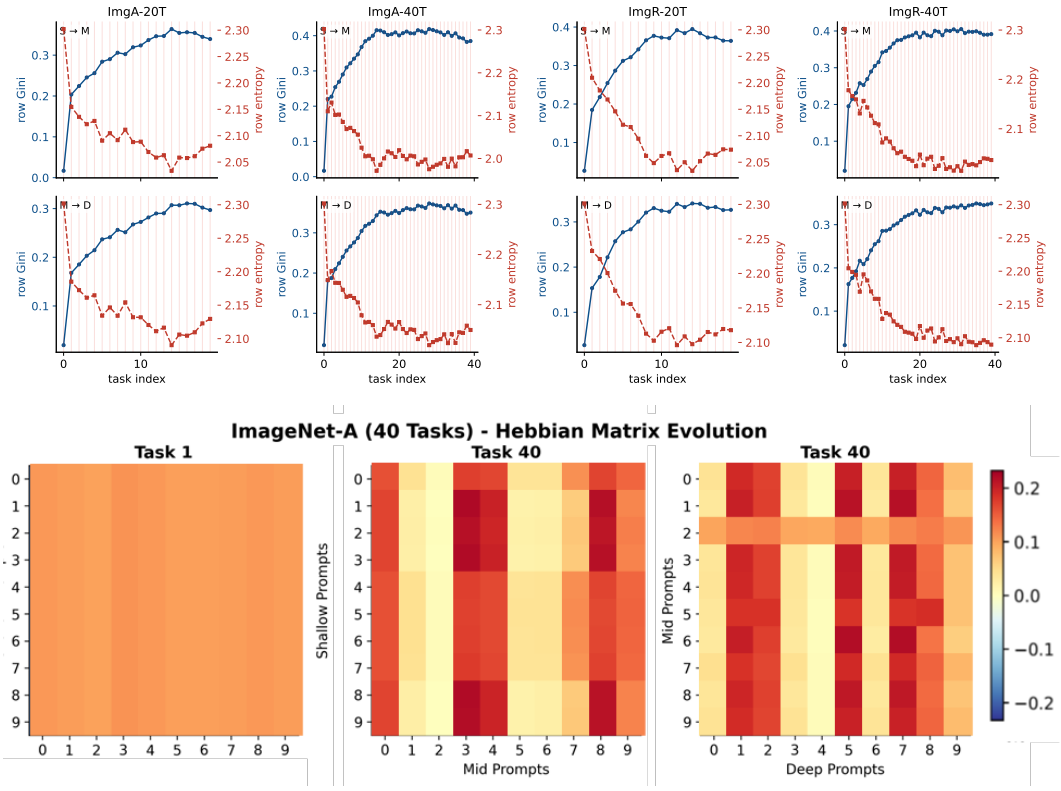


Figure 2: Selection history of L2C. (Top) Gini coefficient and mean row entropy of the transition matrices over the 20- and 40-task sequences of ImageNet-A and ImageNet-R. (Bottom) Shallow→middle after tasks 1 and 40 and middle→deep after task 40 on 40-task ImageNet-A.

shallow prompts connect to a few middle prompts, whereas the middle-to-deep matrix stays more distributed (Section D.8). Many compositions thus pass through a few middle prompts, whose large in-degrees make them hubs (Section D.6), while the deep stage keeps more pathways, in line with its input-specific selection (Section 4.3).

The parameters that L2C adds do not grow with the number of tasks: its prompts, keys, and transition matrices amount to 138,440 parameters, and together with the classifier of the current task to fewer than 0.2% of the backbone. L2C trains at 163.4 images/s, faster than CODA-Prompt (158.9) and at 90% of the throughput of L2P (181.4), which prompts a single stage (Section D.10).

5 Conclusion

L2C addresses the trade-off between allocating parameters to each task and sharing them across tasks by representing knowledge as compositions of prompts selected along the hierarchy of a pre-trained ViT and deciding what to share and protect for each prompt rather than for each task. A selection history that a Hebbian rule accumulates without gradients from the task loss makes this decision: the most used prompts are frozen as hubs shared by later tasks, and the least used ones are re-initialized with keys orthogonal to the hub keys, recycling capacity at a fixed pool size. The analysis shows that this history records the use of each prompt, that its prior widens the selection margin of the compositions in use, and that re-initialized keys are nearly orthogonal to the queries of earlier tasks. In all nine settings on three benchmarks, L2C attains the highest average incremental accuracy and the lowest forgetting, as well as the highest final accuracy on the longest sequence of each benchmark. Because the selection relies on the queries of the frozen backbone, future work will compose prompts on an adapted representation for domains far from the pre-training data, such as VTAB (Section D.1). The Hebbian update uses only the selection distributions, which makes task-free continual learning another direction for future work.

AI USE STATEMENT

Large language models were used solely for grammar correction, final proofreading (checking typos), and verifying notation consistency across the manuscript. The authors verified all text, derivations, code, and results and take full responsibility for the content of the paper.

ETHICS STATEMENT

This work uses only public image-classification benchmarks (CIFAR-100, ImageNet-R, ImageNet-A, and VTAB) under their terms of use and backbones pre-trained on ImageNet; it involves no human subjects and no newly collected data. As with other methods built on these backbones, predictions can reflect biases of the pre-training data. L2C stores no samples or per-sample features of earlier tasks: apart from its prompts, keys, transition matrices, and classifier, it keeps only the mean of the first-task queries, so it applies to settings in which earlier data cannot be retained, such as those restricted by privacy regulations.

REPRODUCIBILITY STATEMENT

Algorithm 1 and Section C specify the procedure, the hyperparameters, and the data splits; Section A states the assumptions and complete proofs of all propositions; and the source code is provided at `https://anonymous.4open.science/r/L2C_ICLR`.

REFERENCES

Keith Ball et al. An elementary introduction to modern convex geometry. *Flavors of geometry*, 31 (1-58):26, 1997.

Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *European conference on computer vision*, pp. 556–572. Springer, 2018.

Cheng Chen, Junchen Zhu, Xu Luo, Heng T Shen, Jingkuan Song, and Lianli Gao. Coin: A benchmark of continual instruction tuning for multimodel large language models. *Advances in neural information processing systems*, 37:57817–57840, 2024.

Shibhansh Dohare, J Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A Rupam Mahmood, and Richard S Sutton. Loss of plasticity in deep continual learning. *Nature*, 632(8026): 768–774, 2024.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

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.

Zhanxin Gao, Jun Cen, and Xiaobin Chang. Consistent prompting for rehearsal-free continual learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 28463–28473. IEEE, 2024.

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.

Haiyang Guo, Fanhu Zeng, Ziwei Xiang, Fei Zhu, Da-Han Wang, Xu-Yao Zhang, and Cheng-Lin Liu. Hide-llava: Hierarchical decoupling for continual instruction tuning of multimodal large language model. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13572–13586, 2025.

Donald Olding Hebb. *The organization of behavior: A neuropsychological theory*. Psychology press, 2005.

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 *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8320–8329. IEEE, 2021a.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15262–15271, 2021b.

Li Jiao, Qiuxia Lai, Yu Li, and Qiang Xu. Vector quantization prompting for continual learning. *Advances in Neural Information Processing Systems*, 37:34056–34076, 2024.

Adrien Journé, Hector Garcia Rodriguez, Qinghai Guo, and Timoleon Moraitis. Hebbian deep learning without feedback. *arXiv preprint arXiv:2209.11883*, 2022.

Gyuhak Kim, Changnan Xiao, Tatsuya Konishi, Zixuan Ke, and Bing Liu. A theoretical study on solving continual learning. *Advances in neural information processing systems*, 35:5065–5079, 2022.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pp. 3519–3529. PMlR, 2019.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Minh Le, An Nguyen, Huy Nguyen, Trang Nguyen, Trang Pham, Linh Van Ngo, and Nhat Ho. Mixture of experts meets prompt-based continual learning. *Advances in Neural Information Processing Systems*, 37:119025–119062, 2024.

Yan-Shuo Liang and Wu-Jun Li. Inflora: Interference-free low-rank adaptation for continual learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23638–23647. IEEE, 2024.

Andreas Maurer, Massi Pontil, and Bernardino Romera-Paredes. Sparse coding for multitask and transfer learning. In *International conference on machine learning*, pp. 343–351. PMLR, 2013.

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.

Martin Menabue, Emanuele Frascaroli, Matteo Boschini, Enver Sangineto, Lorenzo Bonicelli, Angelo Porrello, and Simone Calderara. Semantic residual prompts for continual learning. In *European Conference on Computer Vision*, pp. 1–18. Springer, 2024.

Jorge A Mendez and ERIC EATON. Lifelong learning of compositional structures. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=ADWd4TJO13G`.

Erkki Oja. Simplified neuron model as a principal component analyzer. *Journal of mathematical biology*, 15(3):267–273, 1982.

Oleksiy Ostapenko, Pau Rodriguez, Massimo Caccia, and Laurent Charlin. Continual learning via local module composition. *Advances in Neural Information Processing Systems*, 34:30298–30312, 2021.

Bohao Peng, Zhuotao Tian, Shu Liu, Ming-Chang Yang, and Jiaya Jia. Scalable language model with generalized continual learning. In *International Conference on Learning Representations*, volume 2024, pp. 28412–28434, 2024.

Ameya Prabhu, Hasan Abed Al Kader Hammoud, Puneet K Dokania, Philip HS Torr, Ser-Nam Lim, Bernard Ghanem, and Adel Bibi. Computationally budgeted continual learning: What does matter? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3698–3707, 2023.

Jingyang Qiao, Zhizhong Zhang, Xin Tan, Chengwei Chen, Yanyun Qu, Yong Peng, and Yuan Xie. Prompt gradient projection for continual learning. In *International Conference on Learning Representations*, volume 2024, pp. 16294–16315, 2024.

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.

Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. Progressive prompts: Continual learning for language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=UJTgQBc91_.

Minhyuk Seo, Hyunseo Koh, and Jonghyun Choi. Budgeted online continual learning by adaptive layer freezing and frequency-based sampling. In *International Conference on Learning Representations*, volume 2025, pp. 47499–47533, 2025.

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.

Hai-Long Sun, Da-Wei Zhou, Hanbin Zhao, Le Gan, De-Chuan Zhan, and Han-Jia Ye. Mos: Model surgery for pre-trained model-based class-incremental learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pp. 20699–20707, 2025.

HaiLong SUN, DaWei ZHOU, DeChuan ZHAN, and HanJia YE. Pilot: A pre-trained model-based continual learning toolbox. *Science China Information Sciences*, 68(4), 2023.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pp. 10347–10357. PMLR, 2021.

Liyuan Wang, Jingyi Xie, Xingxing Zhang, Mingyi Huang, Hang Su, and Jun Zhu. Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality. *Advances in Neural Information Processing Systems*, 36:69054–69076, 2023a.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE transactions on pattern analysis and machine intelligence*, 46(8):5362–5383, 2024.

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

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

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.

Yichen Wu, Hongming Piao, Long-Kai Huang, Renzhen Wang, Wanhua Li, Hanspeter Pfister, Deyu Meng, Kede Ma, and Ying Wei. Sd-lora: Scalable decoupled low-rank adaptation for class incremental learning. In *International Conference on Learning Representations*, volume 2025, pp. 58337–58353, 2025.

Mingqing Xiao, Qingyan Meng, Zongpeng Zhang, Di He, and Zhouchen Lin. Hebbian learning based orthogonal projection for continual learning of spiking neural networks. In *International Conference on Learning Representations*, volume 2024, pp. 16271–16293, 2024.

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 *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23219–23230. IEEE, 2024.

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019.

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.

Weixiang Zhao, Shilong Wang, Yulin Hu, Yanyan Zhao, Bing Qin, Xuanyu Zhang, Qing Yang, Dongliang Xu, and Wanxiang Che. Sapt: A shared attention framework for parameter-efficient continual learning of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11641–11661, 2024.

Da-Wei Zhou, Qi-Wei Wang, Han-Jia Ye, and De-Chuan Zhan. A model or 603 exemplars: Towards memory-efficient class-incremental learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=S07feAlQHgM.

Da-Wei Zhou, Hai-Long Sun, Jingyi Ning, Han-Jia Ye, and De-Chuan Zhan. Continual learning with pre-trained models: a survey. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, IJCAI '24, 2024a. ISBN 978-1-956792-04-1. doi: 10.24963/ijcai.2024/924. URL https://doi.org/10.24963/ijcai.2024/924.

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, 2024b.

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, 2025.