THE FEATURES WE MAKE ALONG THE WAY: INTERMEDIATE REPRESENTATIONS FOR TRAINING-FREE CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Intermediate representations can improve frozen-backbone classification, but combining depths, selecting a better layer, and changing the classifier can each contribute. We examine these choices across eleven image datasets and four frozen ViTs, comparing layer selection, normalization, and classifier heads on the same features. On five datasets covering digits, traffic signs, and overhead imagery, normalized all-layer features outperform the best validation-selected single block on 18 of 20 backbone–dataset pairs with KLDA and 19 with RanPAC. Equally wide random nonlinear views of the final block do not recover these gains. Direct ridge regression on raw all-layer features performs best on these datasets, reaching 91.82% accuracy, compared with 88.94% for normalized RanPAC and 88.97% for normalized KLDA. Per-block normalization improves the random-feature heads but leaves ridge accuracy essentially unchanged. With hyperparameters selected without future-task data, all-layer inputs add approximately four percentage points of average incremental accuracy for both random-feature heads on the same five datasets, although restricted baseline tuning likely inflates RanPAC's gain. The dataset grouping was exploratory. A prediction stated before evaluating six held-out datasets separated three MNIST-format datasets from three natural-photograph datasets by relative error reduction under both random-feature heads. A second test separated content from format: the gain on digits survives a change to color photographic images, but converting CIFAR-10 to 28×28 grayscale also produces a gain, so image format contributes as well.

1 INTRODUCTION

A frozen backbone makes continual classification simple to implement: its representation stays fixed, and a classifier can incorporate a new class through accumulated statistics. Nearest-class-mean methods (Janson et al., 2022), RanPAC (McDonnell et al., 2023) and KLDA (Momeni et al., 2025) use this approach with different feature maps and decision rules. Their performance depends on what the backbone exposes to the classifier. A final-layer embedding is convenient, but it is only one possible readout.

Several methods already use intermediate representations. Head2Toe (Evci et al., 2022) selects features across a frozen network for transfer, surgical fine-tuning (Lee et al., 2023) finds that the best layer to adapt depends on the type of distribution shift, and LayUP (Ahrens et al., 2024) combines intermediate states with a second-order classifier for continual learning. LayUP studies depths up to all twelve transformer blocks, uses six as a practical default, and also evaluates multilayer inputs to RanPAC after first-session adaptation. We study how much of the gain comes from combining depths, normalizing their features, and choosing the classifier, with the backbone frozen throughout.

We examine these choices on four frozen ViTs and eleven image datasets. The matched comparison crosses raw and normalized readouts with three heads: ridge regression as in LayUP, RanPAC's random ReLU features, and KLDA's random Fourier features. For the two random-feature heads we also compare all blocks with the best single block chosen on validation data, the last four blocks, and equally wide random transformations of the final block. We call the normalized concatenation a *firing pattern* and use FIRE as shorthand for this readout.

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.

On digits, traffic signs and overhead imagery, combining blocks outperforms selecting one block. All-layer inputs beat the validation-selected single block on 18 of 20 pairs under KLDA and 19 under RanPAC. Random views of the final layer lose on all 40 pairs across the two heads. Ridge regression on the multilayer features gives the highest accuracy. On these datasets, raw all-layer ridge reaches 91.82%, while normalized RanPAC and KLDA reach 88.94% and 88.97%. Normalization helps the random-feature heads but does not improve ridge. The effect of normalization therefore depends on the classifier.

We also evaluate the two random-feature heads along class-incremental sequences, selecting their hyperparameters without future-task data. All-layer inputs add about four points of average incremental accuracy on the same five datasets. Improvements are much smaller on the other six image datasets and on four text benchmarks. We chose the image grouping after seeing the results. A prediction fixed before evaluation separated three MNIST-format datasets from three natural-photograph datasets (§4.7); its applicability beyond those formats remains untested.

Contributions.

- A matched comparison separates depth, normalization and classifier choice. All-layer ridge gives the highest accuracy; on the high-gain datasets, normalization benefits the random-feature heads but leaves ridge unchanged (§4.2).
- Single-block and random-view controls show that, on the high-gain datasets, concatenated blocks outperform every tested alternative: the best single block, the last four blocks and equally wide random views of the final block (§4.5–§4.6).
- A class-incremental evaluation checks whether the random-feature gains survive selection without future-task data, and a per-pair analysis reports their variation across datasets and backbones (§4.6–§4.7).

2 Related Work

Class-incremental learning with pretrained models. A CIL model receives a sequence of tasks that introduce disjoint classes and must discriminate among all classes seen so far without a task identifier at test time. The obstacles are catastrophic forgetting (McCloskey & Cohen, 1989; Kirkpatrick et al., 2017) and the placement of boundaries between classes never observed together. Rehearsal stores exemplars (Rebuffi et al., 2017), prompt-based methods learn small task-conditional parameter sets (Wang et al., 2022b;a), and freezing the backbone rules out representation drift, though not the loss of accuracy on early classes as later ones arrive to compete (Janson et al., 2022; Zhou et al., 2023). RanPAC (McDonnell et al., 2023) combines a fixed nonlinear random projection with decorrelated class prototypes. FeCAM (Goswami et al., 2023) models class-specific covariances in pretrained feature space. RanDumb (Prabhu et al., 2024) instead applies a fixed random feature map to raw pixels in its main online-learning experiments; it separately evaluates linear classifiers on pretrained representations. KLDA (Momeni et al., 2025) applies random Fourier features to frozen representations and fits a shared-covariance discriminant classifier, reporting accuracy comparable to joint training.

Multi-layer readouts and layer-wise analysis. Earlier uses of multilayer features include hypercolumns (Hariharan et al., 2015) which stack per-pixel activations across a convolutional network, and skip connections that combine coarse and fine predictions in dense prediction (Long et al., 2015). Layer-wise analyses provide a related motivation. ELMo learned scalar mixing weights over layers (Peters et al., 2018), Tenney et al. (2019) localized NLP subtasks to characteristic depths, and linear probes (Alain & Bengio, 2016), the tuned lens (Belrose et al., 2023) and sparse probing (Gurnee et al., 2023) all read intermediate states. Closest to our setting, LayUP (Ahrens et al., 2024) established that intermediate-layer representations improve rehearsal-free continual learning: it concatenates raw block states, studies one through twelve blocks, and fits a ridge classifier on their second-order statistics, optionally after adapting the backbone on the first task. LayUP also reports gains from feeding multilayer features from a first-session-adapted backbone to RanPAC's random projection. Head2Toe (Evci et al., 2022) selects features across a frozen network and reports strong out-of-distribution transfer. DINOv2's linear evaluation searches over one or four output layers, with or without average-pooled patch features (Oquab et al., 2024). Our readout normalizes each

block before concatenation, which fixes the scale of the induced distances, reads all blocks with one weighting shared across datasets, and never adapts the backbone. §4.2 compares the two constructions directly. Moser et al. (2026) study layer-dependent regularization during backbone updates, a different use of depth from the frozen readouts studied here. Our exploratory weighting comparison is in Appendix H.

3 METHOD

We define the three classifiers, then the normalized readout and the kernel identity used to construct the controls in §4.

3.1 GRADIENT-FREE CLASSIFIER HEADS

We use *gradient-free* to mean that neither the backbone nor the head is optimized by gradient descent. The head still learns from labels by estimating statistics and solving for a classifier. For an input vector v, the ridge head used by LayUP forms $G = \sum_i v_i v_i^\top$ and $C = \sum_i v_i e_{y_i}^\top$, where e_y is a one-hot class vector. It predicts with $v^\top (G + \lambda I)^{-1} C$. Both G and C are additive, so the head can incorporate new classes without storing earlier examples. We compare this direct readout with two fixed nonlinear feature maps.

RanPAC (McDonnell et al., 2023) takes ϕ to be a fixed random nonlinear projection and fits a ridge-regularized prototype classifier on the decorrelated second-order statistics of ϕ. KLDA (Momeni et al., 2025) takes ϕ to be a random Fourier feature map, which we write out because §3.3 needs its form. Random Fourier features (Rahimi & Recht, 2007) approximate an RBF kernel,

$$\mathbf{z}(\mathbf{x}) = \sqrt{\tfrac{2}{D}}\,\cos\big(\mathbf{\Omega}^\top \mathbf{h}_L(\mathbf{x}) + \mathbf{b}\big), \qquad \mathbf{\Omega} \sim \mathcal{N}(0, 2\gamma I_d), \quad \mathbf{b} \sim \mathrm{Unif}[0, 2\pi], \tag{1}$$

so that $\mathbf{z}(\mathbf{x})^\top \mathbf{z}(\mathbf{x}') \approx \exp(-\gamma \|\mathbf{h}_L(\mathbf{x}) - \mathbf{h}_L(\mathbf{x}')\|^2)$. Classification is linear discriminant analysis in $\mathbb{R}^D$ with one mean $\boldsymbol{\mu}_c$ per class and one covariance $\mathbf{\Sigma}$ shared by all classes,

$$\begin{aligned} \hat{y}(\mathbf{x}) &= \arg\max_c \; \mathbf{z}^\top \mathbf{\Sigma}^\dagger \boldsymbol{\mu}_c - \tfrac{1}{2} \boldsymbol{\mu}_c^\top \mathbf{\Sigma}^\dagger \boldsymbol{\mu}_c, \\ \boldsymbol{\mu}_c &= \frac{1}{N_c} \sum_{i:y_i=c} \mathbf{z}_i, \quad \mathbf{\Sigma} = \sum_c \sum_{i:y_i=c} (\mathbf{z}_i - \boldsymbol{\mu}_c)(\mathbf{z}_i - \boldsymbol{\mu}_c)^\top. \end{aligned} \tag{2}$$

Here $\mathbf{\Sigma}$ is the within-class scatter and $\dagger$ denotes the Moore–Penrose pseudoinverse. Class means and scatter are computed from each complete class batch; equivalently, counts, feature sums and outer-product sums suffice. A new class is absorbed by addition, and for a fixed feature map and fixed hyperparameters the classifier after the last task is the same function of the training set under any task partition or arrival order. RanPAC's accumulators are also class-indexed sums and behave the same way. We write γ for the bandwidth in $\exp(-\gamma\|\cdot\|^2)$ (the reference implementation calls it `sigma`).

3.2 FIRING PATTERNS

Let $\mathbf{h}_l(\mathbf{x}) \in \mathbb{R}^d$ be the pooled state after block l, for $l = 0, \ldots, L$, with $l = 0$ the embedding output and, in encoder–decoder backbones, encoder states preceding decoder states. The firing pattern is

$$\mathbf{p}_\mathbf{w}(\mathbf{x}) = \big[\, w_0 \hat{\mathbf{h}}_0(\mathbf{x}) \;;\; \ldots \;;\; w_L \hat{\mathbf{h}}_L(\mathbf{x}) \,\big] \in \mathbb{R}^{(L+1)d}, \qquad \hat{\mathbf{h}}_l(\mathbf{x}) = \frac{\mathbf{h}_l(\mathbf{x})}{\|\mathbf{h}_l(\mathbf{x})\|_2}, \qquad \|\mathbf{w}\|_2 = 1. \tag{3}$$

The pattern is substituted for $\mathbf{h}_L$ as the argument of ϕ, with the same classifier rule: under KLDA this means Eq. 1 with $\mathbf{\Omega} \sim \mathcal{N}(0, 2\gamma I_{(L+1)d})$ and Eq. 2 unchanged, and under RanPAC a projection of matching input width. The layer weights are shared by every dataset within a modality: uniform, $w_l \propto 1$, on images, and linear, $w_l \propto l + 1$, on text. Both were chosen in early sweeps that read test accuracy (Appendix H); DTD's feature dimension was also test-selected (§4.1). Subsequent bandwidth and ridge selection uses validation data. We use the CLS token for images and mean pooling over non-padding tokens for text.

3.3 KERNEL VIEW

The proofs are in Appendix A. Write $c_l(\mathbf{x}, \mathbf{x}') = \langle \hat{\mathbf{h}}_l(\mathbf{x}), \hat{\mathbf{h}}_l(\mathbf{x}') \rangle \in [-1, 1]$ for the cosine similarity of block l on two inputs, and k_l for the corresponding per-block kernel.

Proposition 1 (Kernel of the firing pattern). *For any* $\mathbf{w}$ *with* $\|\mathbf{w}\|_2 = 1$*, the pattern has unit norm and*

$$K_{\mathbf{w}}(\mathbf{x}, \mathbf{x}') \;=\; \langle \mathbf{p}_{\mathbf{w}}(\mathbf{x}), \mathbf{p}_{\mathbf{w}}(\mathbf{x}') \rangle \;=\; \sum_{l=0}^{L} w_l^2 \, c_l(\mathbf{x}, \mathbf{x}'),$$
$$\|\mathbf{p}_{\mathbf{w}}(\mathbf{x}) - \mathbf{p}_{\mathbf{w}}(\mathbf{x}')\|_2^2 \;=\; 2\big(1 - K_{\mathbf{w}}(\mathbf{x}, \mathbf{x}')\big). \tag{4}$$

Since $w_l^2 \geq 0$ *sum to one,* $K_{\mathbf{w}} = \sum_l w_l^2 k_l$ *is a* convex combination *of the per-block kernels.*

The squared weights form a distribution over blocks. With all mass on the last block, $\mathbf{w} = \mathbf{e}_L$, the kernel is the cosine similarity of the normalized final block. That differs from the KLDA baseline, which feeds the raw final vector to an RBF kernel. A pattern that repeats one block in every slot has $K_{\mathbf{w}} = k_m$ whatever the weights; this is the repeated-block control of §4.5. It matches the pattern's width, per-block norm and projection scale, yet it has the Gram matrix of a single block, and it therefore controls for normalization only. For Gaussian projections the equality holds in distribution over the random draw.

Remark 2 (Random-feature heads). KLDA *applies Eq. 1 to* $\mathbf{p}_{\mathbf{w}}$*, and by Eq. 4 the kernel it approximates is* $\exp(-2\gamma(1 - K_{\mathbf{w}})) = \prod_l \exp(-2\gamma w_l^2(1 - c_l))$*, a product of per-block Gaussian kernels with bandwidths* γw_l^2*. Its log is the* w_l^2*-weighted sum of per-block Gaussian log similarities at the common bandwidth* γ*. Propositions 1 and 4 concern the linear kernel* $K_{\mathbf{w}}$*, not this Gaussian kernel. The exponent lies in* $[-4\gamma, 0]$ *for every backbone, which bounds the range of useful bandwidths independently of activation scale; the bandwidths actually selected for the pattern fall between* 0.2 *and* 0.4 *on every backbone, whereas the baseline's optimum moves with the backbone's activation scale (Appendix E). And since a finite* D *only approximates the kernel, differences between readouts that induce the same kernel are Monte Carlo error.*

The normalized linear kernel also contains each individual block's function space, at a norm cost. Appendix C states this result and its limits; it is not a performance guarantee for the fitted random-feature classifiers.

Cost. The backbone cost is unchanged. The projection $\mathbf{\Omega}$ grows from $D \times d$ to $D \times (L{+}1)d$, a fixed matrix, while the class means and $\mathbf{\Sigma}$ still scale with D, which is therefore the lever on memory (Appendix F). As D approaches the rank of $\mathbf{\Sigma}$, at most $N - |\mathcal{C}|$, inverting its smallest singular values amplifies noise; with the reference solver on DTD, KLDA fell to chance at $D = 1832$ (Appendix V); ridge over the $(L{+}1)d = 9{,}984$-dimensional pattern instead keeps a Gram matrix of that size squared (about 400 MB in float32, 800 MB in the float64 we use) updated at $O((L{+}1)^2 d^2)$ per example, against a 100 MB covariance and $O((L{+}1)dD + D^2)$ per example for all-layer KLDA at $D = 5000$; a sparsification bound is in Appendix B.

4 EXPERIMENTS

4.1 SETUP

Benchmarks. For images we use eleven class-incremental datasets: CIFAR-10 and CIFAR-100 (Krizhevsky, 2009), TinyImageNet (Le & Yang, 2015), and the eight-task transfer suite of Stanford Cars (Krause et al., 2013), DTD (Cimpoi et al., 2014), EuroSAT (Helber et al., 2019), GTSRB (Stallkamp et al., 2011), MNIST (LeCun et al., 1998), RESISC45 (Cheng et al., 2017), SUN397 (Xiao et al., 2010) and SVHN (Netzer et al., 2011). For text we use the four benchmarks of Momeni et al. (2025): CLINC150 (Larson et al., 2019), Banking77 (Casanueva et al., 2020), HWU64 (Liu et al., 2019) and DBpedia (Zhang et al., 2015).

Backbones. Four frozen image backbones span three pretraining paradigms: DINOv2-B/14, self-supervised (Oquab et al., 2024); CLIP ViT-B/16, contrastive image–text (Radford et al., 2021); and ViT-B/16 supervised on ImageNet-21k and on ImageNet-1k (Dosovitskiy et al., 2021). Each

yields 13 CLS states of width 768, taken from the raw hidden states before any final layer norm or projection. Text uses a frozen BART-base (Lewis et al., 2020) with mean pooling, giving 14 states (seven encoder, seven decoder). Per-layer features are extracted once per (dataset, backbone) and cached in float32, so every configuration reads bit-identical inputs.

Heads. We fit ridge regression directly on the readout, following the frozen variant of LayUP. KLDA reads the raw final layer at $D = 5000$. RanPAC (McDonnell et al., 2023) passes features through a fixed random ReLU projection of width $M = 10^4$ (its default) and fits a decorrelated prototype classifier in closed form. We run RanPAC without its first-session adaptation stage, as a purely frozen-backbone head. It is the stronger final-layer random-feature baseline; the matched ridge comparison determines the strongest overall configuration.

Joint-training references. We use two non-incremental reference points. The *joint linear probe* is an ℓ_2-regularized multinomial logistic regression fit on all classes at once on the same cached features by full-batch L-BFGS, with penalty and feature scaling selected on the validation split below. *Joint fine-tuning* updates the whole DINOv2 backbone with a linear head for ten epochs of AdamW, one recipe for every dataset, as in the "upper bound" row of prompt-based CIL (Wang et al., 2022b).

Protocol. Bandwidth and ridge strength are selected using stratified 10% training holdouts, followed by a full-training-set refit. The original image grid uses one selected value per backbone and readout, shared across datasets; the matched and layer-control grids select separately per backbone–dataset–readout. Appendix X records the settings, and Appendix U describes numerical precision. These are validation-selected evaluations of a method developed partly with test feedback, not an untouched confirmatory test. Because this split spans all tasks, data from later tasks helps set the bandwidth and hence the random feature map. This measures final classification accuracy but is not strictly continual; §4.6 repeats the evaluation under a protocol that is. Choosing on the test set instead would have changed one selection (CLIP under KLDA) and moved the KLDA mean by 0.04 points. We use $D = 5000$, a single model on images as in the published image protocol, a five-member ensemble on text, and three random-feature seeds. DTD is the exception: its 1,880 training images over 47 classes bound the rank of KLDA's covariance, and KLDA runs there at $D = 800$, a value set from an earlier sweep that read test accuracy (Appendix V). The text configuration was likewise fixed in an earlier sweep, and the appendix ablations that predate this protocol select γ on the test set; their captions say so.

Evaluation sets. We use two feature extractions. The original grid supports the broad final-layer versus normalized all-layer comparison. The matched comparisons and controls use a second extraction from original releases for CIFAR, TinyImageNet, SVHN, MNIST and GTSRB, and public task-arithmetic mirrors for the other five datasets; the DTD mirror merges train and validation. Each within-table comparison uses identical features and splits. Absolute scores across the two extractions are not interchangeable. For compact tables, obj./scene denotes six object, scene and texture datasets; high-gain denotes five digit, sign and overhead-imagery datasets. The grouping is discussed and tested on held-out datasets in §4.7.

4.2 DEPTH, NORMALIZATION AND CLASSIFIER CHOICE

Table 1 crosses final, last-six and all-layer inputs, raw or normalized, under three heads. Its all-13 raw row is LayUP's all-twelve-block readout plus the constant embedding state, whose removal changes nothing (§4.5). On the high-gain datasets, reading all blocks beats LayUP's last six under every head, and per-block normalization adds 1.69 points under RanPAC and 0.90 under KLDA over raw concatenation (17 and 18 wins out of 20). It adds nothing under the ridge head. On objects, scenes and textures, all readouts beyond the final block are within a point of each other; the one large effect there is a drop for normalized KLDA on Stanford Cars with the ImageNet-1k ViT (40.4 against 55.1 raw), where D sits close to the rank of the covariance; with a float64 solve and a small fixed shrinkage the drop disappears (58.9 against 54.5 raw), so it is numerical. LayUP's ridge readout beats both random-feature heads over the firing pattern: frozen LayUP reaches 89.80 on the high-gain datasets against 88.94 and 88.97, and the ridge head over all thirteen blocks, raw or normalized,

Table 1: Matched comparison on identical features, splits, validation protocol and seeds, over the same backbone–dataset pairs as Table 4. The ridge head fits a regularized least-squares readout on the features themselves, as LayUP does; the raw last-6 row with the ridge head is LayUP without first-session adaptation. Matching cells of Tables 4 and 1 come from two independent runs and agree within 0.16 points; RanPAC and KLDA cells differ in their random projections, ridge cells only in the rule for breaking near-ties in λ.

	ridge (LayUP head)		RanPAC		KLDA	
readout	obj./scene	high-gain	obj./scene	high-gain	obj./scene	high-gain
final block, raw	79.75	82.25	82.20	83.09	79.53	82.23
final block, normalized	79.72	82.08	82.35	83.56	79.81	83.00
last 6, raw (LayUP)	82.95	89.80	82.99	86.71	80.10	87.58
all 13, raw	83.20	**91.82**	**83.11**	87.25	**80.57**	88.07
all 13, normalized (ours)	**83.33**	91.81	82.98	**88.94**	80.09	**88.97**

Table 2: Mean accuracy over eleven class-incremental image benchmarks, by head and readout. Bandwidth and ridge selected on training holdouts; layer weights and DTD D were test-informed. $D = 5000$ (DTD 800); three seeds.

	KLDA		RanPAC	
Backbone	final layer	firing pattern	final layer	firing pattern
DINOv2-B/14 *self-sup.*	84.05	**87.46**	85.63	**88.10**
CLIP ViT-B/16 *contrastive*	85.23	**86.43**	86.62	**87.01**
ViT-B/16 IN-21k *supervised*	76.08	**81.22**	78.59	**82.83**
ViT-B/16 IN-1k *supervised*	77.02	**81.53**	79.71	**83.24**
Mean	80.59	**84.16**	82.64	**85.30**

is the best configuration we measured (87.1 and 87.2 over all 44 pairs, against 85.7 for RanPAC and 84.1 for KLDA over the pattern).

Head and normalization effects. For accuracy on this suite, the evidence favors direct ridge regression on all-layer features. The random-feature heads let us test whether the same information helps under different nonlinear maps; their improvement over final-layer inputs is not a claim of superiority over ridge. Going from the last six to all blocks adds 2.02 points to ridge on the high-gain datasets and 0.25 on the others. Many selections in Table 1 sat on a grid edge, most often for raw inputs under RanPAC. We re-searched every such cell with an extended, scale-aware grid (equal spacing for raw and normalized inputs, extension until the optimum is interior, local refinement, and the stable KLDA solve), after which no selection lies on an edge. Every remaining KLDA cell was then refined to comparable resolution for raw and normalized inputs. The normalization effects on the high-gain datasets remain, $+1.66$ under RanPAC and $+0.62$ under KLDA (19 of 20 pairs), and stay near zero elsewhere (Appendix T). The benefit of combining depths is therefore broader than the benefit of normalization.

Numerical robustness of KLDA. The reference KLDA inverts its covariance with an unregularized float32 pseudoinverse. Repeating its matched cells with a float64 solve after a fixed 10^{-3} shrinkage removes the negative normalization effect on objects, scenes and textures, which came mostly from a numerical collapse on Stanford Cars, and leaves the high-gain effect positive (Appendix T).

4.3 FINAL-LAYER VERSUS ALL-LAYER RANDOM FEATURES

Firing patterns improve KLDA by 3.57 points and are better on 42 of 44 backbone–dataset pairs (Table 2). On the final layer, RanPAC beats KLDA on every backbone, by 2.04 points on average (1.39 to 2.69), and the pattern adds another 2.66 on top of it, less uniformly. RanPAC over the pattern is the best entry in the table at 85.30. Per-dataset results for DINOv2 are in Appendix M: under KLDA the gain runs from $+14.79$ on SVHN down to -0.05 on CIFAR-10. With a frozen BART-base on four text benchmarks the gain is only 0.55 points and is not significant per seed

Table 3: RanPAC against a joint-training reference, a linear probe on the same cached features. Eleven datasets; KLDA columns are in Table 2. Final and pattern denote the final-layer and firing-pattern readouts; gain is their difference in percentage points. The probe is fit on all classes at once.

	RanPAC			joint probe		
Backbone	final	pattern	gain	final	pattern	gain
DINOv2-B/14	85.63	**88.10**	+2.47	85.79	**88.65**	+2.86
CLIP ViT-B/16	86.62	**87.01**	+0.39	86.65	**87.56**	+0.91
ViT-B/16 IN-21k	78.59	**82.83**	+4.24	79.05	**83.10**	+4.05
ViT-B/16 IN-1k	79.71	**83.24**	+3.53	79.52	**83.27**	+3.75
Mean	82.64	**85.30**	+2.66	82.75	**85.64**	+2.89

(Appendix K). Table 2 uses the reference KLDA solve; with a stabilized solve the matched KLDA cells move by at most 1.4 points on average and the normalization effect keeps its sign on the high-gain datasets (Appendix T).

Significance. We test every pair with an exact McNemar test per seed (Table 11). Under KLDA, 36 of 44 pairs win on all three seeds and one loses. Under RanPAC, 27 win and 3 lose: DINOv2 on TinyImageNet and CLIP on Stanford Cars and SUN397. For a pair-level claim that holds across seeds, we use the maximum of its three per-seed p-values and apply Holm correction across the 44 pairs. This retains 32 wins and one loss under KLDA, and 22 wins and two losses under RanPAC. The bootstrap intervals on the mean gain exclude zero but are wide, $[+1.49, +6.34]$ for KLDA and $[+0.66, +5.35]$ for RanPAC (per-dataset gains range from -4.3 to $+14.8$). The averages are unweighted over datasets, and sample-weighted means would be higher because SVHN and GTSRB have the largest test sets. They also lean on the two ImageNet-supervised backbones: over DINOv2 and CLIP alone, the gains are 2.31 (KLDA) and 1.43 (RanPAC). Seeds vary only the random-feature draw, so seed spreads understate uncertainty, and we rely on the paired tests.

4.4 Comparison with joint training

On the final layer, the joint probe lands within 0.5 points of RanPAC on every backbone (Table 3). The two classifiers therefore obtain similar accuracy on this readout. The probe gains 2.89 points from the firing pattern, whose RanPAC score is 0.34 points below the probe. The probe is linear and regularized, and it is not a ceiling for the nonlinear heads: on GTSRB with DINOv2, KLDA over the pattern beats it by 3.9 points.

Joint fine-tuning of every DINOv2 weight reaches 93.65 on average (Table 12, Appendix M). The gap between it and the frozen final-layer KLDA head is 15 points on the high-gain datasets but only 5 on objects, scenes and textures. As a ratio of mean accuracies, the pattern closes 41% of the first gap (53% if per-dataset fractions are averaged, and 20% of the second gap on the same basis).

4.5 Normalization and width controls

FIRE normalizes each block, is thirteen times wider than the baseline, and includes the blocks below L. Repeating the normalized final block in every slot matches the pattern's width and per-block norms, but under a Gaussian random projection it is equal in distribution to the normalized final block at native width (Proposition 1). It therefore tests normalization alone. On the original features this control averages 79.96 against the 80.59 baseline (Table 8, Appendix I), and the full pattern is 4.20 points above it. On the second feature extraction, normalizing the final block moves KLDA by +0.3 on objects, scenes and textures and +0.8 on the high-gain datasets (Table 1). Either way the effect is small next to that of the full pattern.

To test width, we fill the twelve extra slots with distinct nonlinear random views of the final block, $\mathrm{relu}(\mathbf{h}_L R_l)$ with independent Gaussian R_l, each normalized like a real block. This input is as wide and as varied as the pattern, but every coordinate is a function of $\mathbf{h}_L$. On the five high-gain datasets it scores 82.41 under KLDA and 82.96 under RanPAC, against 88.92 and 89.10 for the real pattern, and it loses on all 40 backbone–dataset–head combinations (Table 4). Across all eleven datasets the real

Table 4: Controls, grouped as in §4.7: objects, scenes and textures (CIFAR-10, CIFAR-100, TinyImageNet, Stanford Cars, DTD, SUN397) and the high-gain group (SVHN, GTSRB, MNIST, RESISC45, EuroSAT). *Top:* test accuracy of the best single block chosen on validation data, the last four blocks, thirteen random views of the final block (§4.5), and all thirteen blocks, all normalized and each with its own validation-selected hyperparameter, three seeds. *Bottom:* average incremental accuracy under the strictly continual protocol, five class orders. Features from the second extraction described in §4.1.

head	group	best single	last 4	rand. views	all blocks
ridge	obj./scene	79.70	82.28	—	**83.31**
	high-gain	84.70	87.84	—	**91.83**
RanPAC	obj./scene	82.32	82.89	82.00	**82.95**
	high-gain	84.31	86.73	82.96	**89.10**
KLDA	obj./scene	79.81	**80.35**	79.18	80.10
	high-gain	85.07	86.79	82.41	**88.92**

head	group	final layer	all blocks
ridge (raw)	obj./scene	85.59	**87.86**
	high-gain	87.85	**93.95**
RanPAC (pattern)	obj./scene	87.11	**87.78**
	high-gain	88.06	**92.49**
KLDA (pattern)	obj./scene	84.04	**84.84**
	high-gain	87.24	**91.14**

pattern wins on 81 of 88. In these controls and in the single-block sweep, the selected ridge strength is interior to its grid in all but one of 308 ridge and RanPAC cells; KLDA's five-point bandwidth grid is hit at an edge in 25 to 65% of cells, so its control numbers carry that caveat (Appendix T). The embedding-layer CLS state (block 0) is identical for every image; dropping it changes the all-block result by at most 0.25 points on average under either head. Because that constant state could act as an intercept for ridge, which has none, we also fit ridge with an explicit, unpenalized per-class intercept: on the high-gain datasets the final block, the last six blocks, the twelve transformer blocks and all thirteen states then score 82.3, 89.8, 91.8 and 91.8, within 0.1 of the scores without an intercept.

Canonical final embeddings. The final-layer rows use the raw last hidden state, before the backbone's final LayerNorm. If instead the final-layer baseline is chosen per pair and head on validation data among the raw and post-LayerNorm final states, each raw or normalized, it improves by at most about one point. All blocks still beat it on all 20 high-gain pairs under every head, by 9.6 points for ridge, 5.3 for RanPAC and 5.8 for KLDA, and by 3.4, 0.5 and 0.3 on objects, scenes and textures.

4.6 LAYER SELECTION AND STRICTLY CONTINUAL EVALUATION

A pattern could beat the final layer simply because one intermediate block is better than the last. We therefore sweep every single block through the same normalization, head and validation protocol (Table 4, top). On objects, scenes and textures, validation picks the final block every time, and all thirteen blocks are no better than the last four (-0.25 points under KLDA, $+0.06$ under RanPAC). On the high-gain datasets, validation instead picks an intermediate block on 17 of 20 pairs under KLDA and 18 of 20 under RanPAC, so here a single intermediate block does beat the final one. All thirteen blocks are better still: they beat the best single block on 18 (KLDA) and 19 (RanPAC) of 20 pairs, and they beat the last four on all 20 under both heads, by 2.1 and 2.4 points on average and by more than six on SVHN. The main protocol lets later tasks influence the bandwidth, so we also use a strictly continual protocol whose selection uses no future-task data. KLDA's bandwidth and D are fixed from the first task alone; ridge and RanPAC re-select their ridge at each step on the incoming task's validation rows, which their running Gram matrices permit (details and task splits in Appendix Q). On the high-gain datasets all-layer inputs add about four points of average incremental accuracy under both random-feature heads (Table 4, bottom), and about five if MNIST, whose two-class first task leaves KLDA's bandwidth unidentified, is left out; on objects, scenes and textures they add 0.8 and 0.7. Direct ridge over all raw blocks is again strongest, at 93.95 and 87.86. Ridge's

per-step selections are interior in 99% of steps for all blocks and 93% for the final layer. RanPAC's final-layer baseline, by contrast, selects a ridge on an edge of the grid in about 60% of steps, while its all-block input almost never does, so RanPAC's strict-protocol gain is likely overstated.

4.7 Variation across datasets

The transfer results of Evci et al. (2022) and Ahrens et al. (2024) motivate examining how the gains vary across datasets. We describe that variation by image content (Table 9, Appendix J); we do not measure each dataset's distance from pretraining. Six of the eleven datasets concern objects, scenes or textures: CIFAR-10, CIFAR-100, TinyImageNet, Stanford Cars, SUN397 and DTD. The other five are street and handwritten digits, traffic signs and overhead imagery. On these five the readout removes 0.41 of the baseline's errors under KLDA and 0.34 under RanPAC, against 0.06 and 0.01 on the first group, where all three uncorrected RanPAC losses also fall. Under KLDA, relative error reduction is larger for every one of the 20 pairs in the second group than for any of the 24 in the first (ranges $[0.15, 0.66]$ and $[-0.17, 0.12]$). Under RanPAC, 465 of the 480 cross-group comparisons agree. We chose the grouping after seeing the results. Its separation depends on the placement of DTD: moving this texture dataset to the second group would remove the separation under KLDA. The groups also overlap in absolute accuracy gains, because relative error reduction magnifies small changes near ceiling (on DINOv2, DTD gains 2.31 points and MNIST 0.70). These comparisons describe where multilayer readouts help in this suite, without identifying the cause. The text benchmarks provide little evidence of a comparable gain.

Held-out test. Before evaluating six additional datasets, we wrote down a prediction based on the exploratory grouping (Appendix S). KMNIST, EMNIST-letters and Fashion-MNIST, which are not natural photographs, should behave like the high-gain group; STL-10, Oxford-IIIT Pets and Oxford Flowers-102 should behave like the others. All three criteria stated in advance held. Under KLDA the first three remove 0.44, 0.31 and 0.16 of the baseline's errors and the last three 0.03, -0.06 and 0.07; under RanPAC the figures are 0.48, 0.32, 0.17 against -0.02, -0.17, -0.05. All three predicted-high datasets are 28×28 grayscale images in the MNIST format. The original suite contains gains on color images, including SVHN, traffic signs and overhead imagery, so the observed benefit is not confined to grayscale inputs. However, those datasets were used to form the grouping and cannot independently validate its explanation. The held-out test establishes a separation between these three MNIST-format datasets and these three photographic datasets.

Boundary test. A second prediction, also written before the experiments, targeted this confound with four new cases scored against the best validation-selected final embedding (Appendix S). On MNIST-M, digits on color photographic patches, and on USPS, all blocks remove 0.43 to 0.67 and 0.34 to 0.44 of the errors across the three heads, as predicted. CIFAR-10 and STL-10 converted to 28×28 grayscale were predicted flat; STL-10 is, but CIFAR-10 gains 0.13 to 0.28. The gain therefore does not require MNIST-like images, but a change to low-resolution grayscale can also create one.

Both random-feature heads lose with CLIP on Stanford Cars (-2.57 under KLDA, -4.27 under RanPAC). The joint probe on CLIP's pattern is also 3.4 points below the probe on its final layer. The appendix reports bandwidth transfer across backbones, the accuracy–memory trade-off, backbone and expert fusion, and layer weighting and sparsification; all but the first select γ on the test set and are exploratory. None is needed for the matched head and depth comparisons.

5 Conclusion

The largest gains from intermediate representations survive controls for normalization, width, single-block selection, the backbone's final LayerNorm and a matched intercept, and persist when hyperparameters are selected without future-task data. Ridge on all-layer features is the strongest readout we measured; per-block normalization matters mainly for random-feature heads. In tests on ten held-out cases with predictions stated in advance, digits gained in every format tested, all at 28 pixels or less; natural photographs at native resolution did not, but CIFAR-10 converted to 28-pixel grayscale did.

Reproducibility Statement

The readouts are specified in Eq. 3, and the classifiers in §3.1. Appendix X gives checkpoint identifiers, numerical choices and selection settings. Comparisons within a feature extraction use the same cached float32 features; the original and matched grids use different extractions, as described in §4.1. For fixed features and hyperparameters, the sufficient-statistic classifiers are order-invariant in exact arithmetic. The strict-protocol evaluation reconstructs these classifiers at each prefix and varies five class orders. Code, the exact command lines, the split manifest of the second extraction and both written predictions will be released upon publication. The remaining provenance and protocol limitations are stated in Appendix D.

Ethics Statement

This work studies image and text classification on public benchmarks and collects no new data. Frozen-backbone continual learners inherit the biases of their backbone with no mechanism to correct them. Reading different layers of the same model should not be expected to remove those biases.

AI Use Statement

We used generative AI tools, LLM-based coding and writing assistants, for the following tasks with required disclosure: implementing methods and experiment code; designing experiments and giving feedback on them, interpreting results, including drafting the text that reports them, formulating mathematical claims and assisting in the writing of proofs. We did not use them to generate synthetic data, to clean or reformat datasets, for translation, or for qualitative data analysis. For tasks with recommended disclosure, we used them to create and edit software code, to suggest experimental parameters, to draft and edit parts of the paper for readability, to format references, to search for related work and dataset sources (all cited papers were read and verified by the authors), and to propose the title. All AI-assisted work was reviewed by the authors. Every reported number was produced by code executed on the authors' infrastructure and checked against the logged outputs, and the incremental evaluation code (§4.6) was checked against a brute-force implementation before use. We take responsibility for the full content of this paper, including text, claims and artifacts produced with the aid of generative AI.

References

Kyra Ahrens, Hans Hergen Lehmann, Jae Hee Lee, and Stefan Wermter. Read between the layers: Leveraging multi-layer representations for rehearsal-free continual learning with pre-trained models. *Transactions on Machine Learning Research*, 2024.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

Nora Belrose, Zach Furman, Logan Smith, Danny Halawi, Igor Ostrovsky, Lev McKinney, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023. URL https://arxiv.org/abs/2303.08112v1.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on NLP for Conversational AI*, 2020.

Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *International Conference on Artificial Intelligence and Statistics*, 2011.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. EMNIST: Extending MNIST to handwritten letters. In *International Joint Conference on Neural Networks*, 2017.

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 (ICLR)*, 2021.

Utku Evci, Vincent Dumoulin, Hugo Larochelle, and Michael C. Mozer. Head2toe: Utilizing intermediate representations for better transfer learning. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 6009–6033, 2022.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.

Dipam Goswami, Yuyang Liu, Bartłomiej Twardowski, and Joost van de Weijer. Fecam: Exploiting the heterogeneity of class distributions in exemplar-free continual learning. In *Advances in Neural Information Processing Systems*, 2023.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding neurons in a haystack: Case studies with sparse probing. *Transactions on Machine Learning Research (TMLR)*, 2023.

Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. EuroSAT: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

Jonathan J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *International Conference on Learning Representations*, 2023.

Paul Janson, Wenxuan Zhang, Rahaf Aljundi, and Mohamed Elhoseiny. A simple baseline that questions the use of pretrained-models in continual learning. *arXiv preprint arXiv:2210.04428*, 2022.

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

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3D object representations for fine-grained categorization. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2013.

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

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of EMNLP-IJCNLP*, 2019.

Ya Le and Xuan Yang. Tiny ImageNet visual recognition challenge. Technical report, Stanford University, 2015.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yoonho Lee, Annie S. Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. Surgical fine-tuning improves adaptation to distribution shifts. In *International Conference on Learning Representations*, 2023.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. Benchmarking natural language understanding services for building conversational agents. In *International Workshop on Spoken Dialogue Systems Technology (IWSDS)*, 2019.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165, 1989.

Mark D. McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton van den Hengel. RanPAC: Random projections and pre-trained models for continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947.

Saleh Momeni, Sahisnu Mazumder, and Bing Liu. Continual learning using a kernel-based method over foundation models. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 39, pp. 19528–19536, 2025.

Brian B. Moser, Ahmed Anwar, Tobias Christian Nauen, Shishir Muralidhara, Federico Raue, René Schuster, Stanislav Frolov, and Andreas Dengel. Layers matter: Why continual learning regularization should be layer-adaptive. *arXiv preprint arXiv:2608.15901*, 2026.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research (TMLR)*, 2024.

Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, 2018.

Ameya Prabhu, Shiven Sinha, Ponnurangam Kumaraguru, Philip H. S. Torr, Ozan Sener, and Puneet K. Dokania. Randumb: Random representations outperform online continually learned representations. In *Advances in Neural Information Processing Systems*, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, 2021.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2007.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The German traffic sign recognition benchmark: A multi-class classification competition. In *International Joint Conference on Neural Networks (IJCNN)*, 2011.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. DualPrompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision (ECCV)*, 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022b.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

Da-Wei Zhou, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. *arXiv preprint arXiv:2303.07338*, 2023. URL https://arxiv.org/abs/2303.07338v1.

A PROOFS

Proposition 1.

Proof. Blocks occupy disjoint coordinates, so $\langle \mathbf{p}_{\mathbf{w}}(\mathbf{x}), \mathbf{p}_{\mathbf{w}}(\mathbf{x}') \rangle = \sum_l w_l^2 \langle \hat{\mathbf{h}}_l(\mathbf{x}), \hat{\mathbf{h}}_l(\mathbf{x}') \rangle$ and $\|\mathbf{p}_{\mathbf{w}}(\mathbf{x})\|_2^2 = \sum_l w_l^2 = 1$; the distance identity follows by expanding. □

Proposition 4.

Proof. $K_{\mathbf{w}} - w_m^2 k_m = \sum_{l \neq m} w_l^2 k_l$ is positive semi-definite, so $w_m^2 k_m \preceq K_{\mathbf{w}}$, which gives $\mathcal{H}_{w_m^2 k_m} \subseteq \mathcal{H}_{K_{\mathbf{w}}}$ with $\|f\|_{K_{\mathbf{w}}} \leq \|f\|_{w_m^2 k_m}$ (Aronszajn, 1950). Rescaling a kernel by $a > 0$ scales the squared norm by $1/a$, so $\|f\|_{w_m^2 k_m} = \|f\|_{k_m} / w_m$. Unit norm gives $K_{\mathbf{w}}(\mathbf{x}, \mathbf{x}) = 1$. □

Proposition 5.

Proof. $E_m(\mathbf{p}_\mathbf{w}(\mathbf{x})) = \hat{\mathbf{h}}_m(\mathbf{x})$, so every $g \in \mathcal{G}$ is matched by $g \circ E_m \in \mathcal{F}$ attaining the same risk. The infimum on the left is over a set that is no smaller. □

Proposition 3.

Proof. Write $\hat{\mathbf{h}}_l = a + b$ with a supported on S_l and b on its complement, so $\|a\|^2 = 1 - \varepsilon_l$ and $a \perp b$. The renormalized block is $a/\|a\|$, and $\|\hat{\mathbf{h}}_l - a/\|a\|\|^2 = 2 - 2\|a\| = 2(1 - \sqrt{1-\varepsilon_l})$. Summing over disjoint blocks with weights w_l^2 gives the equality, and $1 - \sqrt{1-\varepsilon} \le \varepsilon$ on $[0,1]$ gives the bound. The kernel statement follows by adding and subtracting $\langle \mathbf{p}_S(\mathbf{x}), \mathbf{p}_\mathbf{w}(\mathbf{x}')\rangle$ and applying Cauchy–Schwarz, both codes having unit norm; the last claim is Cauchy–Schwarz again. □

B Sparsification Bound

The pattern is $(L{+}1)$ times wider than the embedding it replaces, and Appendix H keeps only a fraction τ of each block's coordinates. Let S_l be the retained support of block l and let

$$\varepsilon_l(\mathbf{x}) \;=\; \big\|\hat{\mathbf{h}}_l(\mathbf{x})\big|_{S_l^c}\big\|_2^2, \qquad \bar{\varepsilon}(\mathbf{x}) \;=\; \sum_l w_l^2\, \varepsilon_l(\mathbf{x}) \tag{5}$$

be the fraction of block l's squared activation mass that selection discards, and its weighted mean. Given a support, both are read off the features alone; choosing the support may itself use labels, and the class-level rule of Appendix H does.

Proposition 3 (Sparsification bound)**.** *Let $\mathbf{p}_S(\mathbf{x})$ be the sparsified pattern with each surviving block renormalized to unit length, and assume $\varepsilon_l(\mathbf{x}) < 1$ for every l, so that no retained block is identically zero. Then*

$$\|\mathbf{p}_\mathbf{w}(\mathbf{x}) - \mathbf{p}_S(\mathbf{x})\|_2^2 \;=\; \sum_l w_l^2\, 2\big(1 - \sqrt{1-\varepsilon_l(\mathbf{x})}\big) \;\le\; 2\,\bar{\varepsilon}(\mathbf{x}). \tag{6}$$

Consequently $|K_\mathbf{w}(\mathbf{x},\mathbf{x}') - K_S(\mathbf{x},\mathbf{x}')| \le \sqrt{2\bar{\varepsilon}(\mathbf{x})} + \sqrt{2\bar{\varepsilon}(\mathbf{x}')}$, and every linear functional $\mathbf{u} \mapsto \langle \boldsymbol{\theta}, \mathbf{u}\rangle$ with $\|\boldsymbol{\theta}\|_2 \le B$ changes by at most $B\sqrt{2\bar{\varepsilon}(\mathbf{x})}$ between the two codes.

The statement concerns the code and its linear functionals; a particular head may depend on the features nonlinearly and through fitted statistics. The bound only works in one direction: it certifies that a support preserving most of the activation mass moves the code little, and $\bar{\varepsilon}$ can be computed before any classifier is fitted, so a sparsity level can be audited in advance. It does not identify a good support, and the experiments show that it cannot: per-image magnitude selection minimizes $\bar{\varepsilon}(\mathbf{x})$ at every input by construction, yet loses to a class-level criterion with strictly larger dropped mass, by 1.76 points at $\tau = 0.25$ (Appendix H).

C Linear-Kernel Containment and Population Risk

The reproducing kernel Hilbert space (RKHS) $\mathcal{H}_{K_\mathbf{w}}$ makes the benefit and the cost of this combination explicit.

Proposition 4 (RKHS containment)**.** *Let $w_m > 0$. Then $\mathcal{H}_{k_m} \subseteq \mathcal{H}_{K_\mathbf{w}}$, and every $f \in \mathcal{H}_{k_m}$ satisfies*

$$\|f\|_{K_\mathbf{w}} \;\le\; \|f\|_{k_m} \,/\, w_m. \tag{7}$$

Under uniform weights $1/w_m = \sqrt{L{+}1}$. Moreover $K_\mathbf{w}(\mathbf{x},\mathbf{x}) = k_l(\mathbf{x},\mathbf{x}) = 1$ for every $\mathbf{x}$ and every l.

Within this family, reading every block loses nothing that the normalized final block could express, at a norm cost bounded by Eq. 7. The result concerns normalized linear kernels only.

Proposition 4 compares function classes, while a head is judged by the risk it achieves. Extending the result to risk requires one explicit assumption, since containing the affine maps is not enough on its own: the family must be closed under *precomposition* with the embedding that isolates a block.

Proposition 5 (Population risk). *Let $\mathcal{F}$ be a set of predictors on $\mathbb{R}^{(L+1)d}$, let $w_m > 0$, and let $E_m(\mathbf{u}) = \mathbf{u}_m / w_m$ read off block m and rescale it. Suppose $g \circ E_m \in \mathcal{F}$ for every predictor g on $\mathbb{R}^d$ in the corresponding block-level family $\mathcal{G}$. Then for any loss ℓ,*

$$\inf_{f \in \mathcal{F}} \mathbb{E}\, \ell\big(f(\mathbf{p}_{\mathbf{w}}(X)), Y\big) \;\le\; \inf_{g \in \mathcal{G}} \mathbb{E}\, \ell\big(g(\hat{\mathbf{h}}_m(X)), Y\big). \tag{8}$$

The assumption holds for linear and affine families, including the linear probe of §4.4 in its unregularized limit. For a kernel class it has to be checked for the kernel at hand, since $g \circ E_m$ need not lie in the class, and we verify it only for the normalized linear kernels of Proposition 4. A norm constraint breaks the assumption, because $g \circ E_m$ carries the inflated norm of Eq. 7 and may fall outside the ball; that is one available explanation for a regularized probe scoring below the final layer on a given dataset, as it does for CLIP on Stanford Cars. It also fails for a head that does not minimize over a family at all. Nearest class mean sets its centroids by averaging, so Eq. 8 says nothing about it and depth may hurt.

D Limitations

Every image backbone here is base-scale, with 12 transformer blocks and an embedding state, so we cannot say whether the gain grows with depth. We run RanPAC and LayUP without their first-session adaptation stages, and we do not test whether adaptation already recovers what the intermediate blocks provide; first-session adaptation could change the relative value of intermediate features. Our benchmarks come from the task-arithmetic suite rather than the standard pretrained-model CIL suites (ImageNet-R, ImageNet-A, CUB-200, OmniBenchmark, VTAB), so our numbers are not comparable to published CIL tables. RanPAC's projection width stays at $M = 10^4$, which barely exceeds the $9{,}984$-dimensional pattern; a wider projection may help it. Under RanPAC the readout loses on 3 of 44 pairs (Table 11). The expert bank is compared only against oracle routing, not against majority vote or max-confidence routing. The strictly continual protocol's first-task bandwidth selection is unreliable when the first task has two classes. The dataset grouping was chosen after observing the original results. The boundary test shows that format contributes as well as content, and it covers four cases. A measure of distance from pretraining is a possible predictor to test; we have not established that it predicts the gains.

E Bandwidth Transfer

Table 5: Bandwidth selected on the validation split per backbone, as used in Table 2. KLDA's optimum tracks the final-layer norm; the firing pattern has unit norm, so its optimum stays put.

Backbone	$\|\mathbf{h}_L\|$	KLDA $\gamma^\star$	FIRE $\gamma^\star$
CLIP ViT-B/16	7.87	7.4×10^{-3}	0.4
DINOv2-base	37.08	1.5×10^{-4}	0.2
ViT-B/16 IN-1k	187.91	6.9×10^{-6}	0.2
ViT-B/16 IN-21k	198.21	6.9×10^{-6}	0.2
Range	25×	1070×	**2**×

Proposition 1 puts every pattern's distances in $[0, 4]$, so one γ near unity is well scaled for any backbone. Table 5 shows the consequence: the baseline's optimum follows the final-layer norm across the four backbones and moves by three orders of magnitude, while ours moves by a factor of two. Fixing $\gamma = 0.2$ for all four backbones and all eleven datasets costs at most 0.35 points. The published KLDA settings span 5×10^{-6} to 5×10^{-2} across its own backbones, the same phenomenon at wider scope.

This shows that the bandwidth transfers across backbones; sensitivity to a misspecified bandwidth is about the same for both methods. Over a $100\times$ span centered on each method's own optimum, the baseline's mean accuracy ranges over 40.78 points and FIRE's over 38.29, so a given multiplicative misspecification of γ hurts both about equally. Comparisons on a shared absolute grid suggest otherwise only because the two methods operate at different scales.

F ACCURACY AND MEMORY

A firing pattern is 13 times wider than the embedding it replaces, which multiplies the projection $\mathbf{\Omega}$ by 13 and leaves the $D \times D$ covariance and the class means unchanged. Both terms scale with D, so D controls memory, and changing it affects both kernel approximation and covariance conditioning (accounting in Appendix O). Firing patterns at $D = 1000$ beat KLDA at $D = 5000$ on accuracy while using $0.84\times$ the memory, and at $D = 2000$ they gain 2.08 points at $0.96\times$. Both totals count the frozen backbone, which must be resident to produce features and at 330 MB is the larger term until D exceeds about 3000; counting the classifier alone would give $0.38\times$ at $D = 1000$. We count $\mathbf{\Omega}$ as stored; it could instead be regenerated from its seed. Both methods are still improving at $D = 5000$, though with sharply diminishing returns: the baseline gains 3.58 points from $D = 500$ to $D = 5000$ for $43\times$ the classifier memory.

G BACKBONE AND EXPERT FUSION

Table 6: Depth against breadth. Eleven datasets, KLDA head, fusing CLIP and DINOv2, the two backbones not trained with ImageNet labels. Per-cell test-selected γ (earlier protocol).

	one backbone	CLIP + DINOv2
final layer	84.05	88.97
all layers	87.43	**89.59**

Table 7: Eight from-scratch ResNet-18 experts on the joint 758-class problem, no router. Task-hit is how often the prediction lands in the correct task's class block.

Configuration	accuracy	task-hit
oracle routing	81.78	100.00
all stages, fused	**81.22**	94.88
final stage only	80.50	93.76
chance	0.13	—

Reading every layer of one backbone varies depth; concatenating several backbones varies breadth, and Table 6 varies both. Adding depth improves accuracy by 3.38 points with one backbone and 0.62 points with two, winning on 6 of 11 datasets in the latter setting. Adding the second backbone improves accuracy by 4.92 points for final-layer inputs and 2.16 points for all-layer inputs. Against an oracle that picks the better backbone per dataset (89.06), final-layer fusion is 0.09 points behind and the full fused pattern 0.53 ahead; that oracle chooses the backbone by test accuracy, so the comparison we report is against the best fixed backbone. Fusion does not reduce memory, since two resident encoders impose a 659 MB floor (Appendix O), and a third and fourth backbone add nothing further.

Expert bank. The construction does not require pretrained backbones. Treating the eight transfer datasets as one problem of 758 classes, with all eight experts available from the start (so the bank itself does not grow incrementally), we trained one ResNet-18 per task from scratch on that task's data alone, fused all eight into a single firing pattern, and classified with the shared-covariance head, without a gate at inference. For any test image, seven of the eight blocks are out of distribution. The result is 0.56 points below oracle routing (Table 7) while choosing among all 758 classes, and the prediction falls inside the correct task's class block 94.88% of the time without a task label, which is what an explicit router would have to get right. Depth adds $+0.72$ here too. The bank costs 94M parameters and 14.4 GFLOP per image against a single ViT-B/16's 87M and 17.6 GFLOP, $0.82\times$ the inference cost of one pretrained backbone. Per-task results are in Appendix P; the gains concentrate where the dedicated expert is weakest, DTD at $+2.77$, and the two losses occur on tasks where the individual expert is already strong.

Incremental behavior. The joint problem also lets us check order invariance empirically. Over 20 task orders the final accuracy is 81.22 at both its minimum and its maximum. Average incremental accuracy is 80.99 ± 8.94, and that spread belongs to the benchmark: first-step accuracy is linearly associated with the average ($r = 0.835$, $r^2 = 0.70$ across the 20 orders), from 32.2 starting with DTD to 99.8 starting with MNIST. Depth helps the whole curve, 80.99 against 80.25 for the final stage alone. A suite that unions tasks from 10 classes at 99.7% to 47 classes at 27% disperses average incremental accuracy in a way homogeneous benchmarks such as Split-CIFAR-100 do not, which limits its comparability across suites.

H LAYER WEIGHTING AND SPARSIFICATION

Three settings offered a natural data-driven weighting and none beat a uniform one (Appendix N). Over layers, five depth schedules including a measured Fisher profile all lose to uniform, the decreasing ones badly: they spend their largest share on layer 0 of a ViT, the CLS token before any attention block, whose variance across images is zero. No weighting over backbones improved on an even blend. We tested expert weighting because individual experts ranged from 27.4% to 99.7% in-task accuracy. Weighting by in-task accuracy (81.10), training-set size (80.85) or class count (80.13) scored below uniform weighting (81.22).

Accuracy rises monotonically with the fraction of each block's units retained. Neither tested sparsification rule matched the dense readout at the evaluated fractions below full retention. At 25% retention, class-level Fisher selection based on training statistics remains 1.42 points above the final-layer baseline; at 10%, it falls 1.17 points below it. A support chosen at the class level beats one chosen per image at every sparsity, by 1.76 points at a quarter, possibly because per-image magnitude selection favors the same large-magnitude coordinates across inputs; we did not measure support overlap directly.

I WIDTH-MATCHED CONTROL RESULTS

Table 8: Control. *Final block* $\times 13$ repeats the ℓ_2-normalized final block in all thirteen slots, which matches the firing pattern's width and per-block norms and, by Proposition 1, induces the kernel of the normalized final block alone. Eleven datasets, KLDA head, validation protocol.

Backbone	KLDA (final, raw)	final block $\times 13$	all 13 blocks
DINOv2	84.05	81.69	**87.46**
CLIP	85.23	85.04	**86.43**
ViT-21k	76.08	75.78	**81.22**
ViT-1k	77.02	77.32	**81.53**
Mean	80.59	79.96	**84.16**

J RELATIVE ERROR REDUCTION PER DATASET

Table 9: Fraction of the baseline's test errors removed by the firing pattern, $(e_{\text{base}} - e_{\text{FIRE}})/e_{\text{base}}$, per dataset, averaged over the four backbones, under each head. The first group contains object, scene and texture datasets; the second contains digits, traffic signs and overhead imagery.

			fraction of errors removed	
Dataset	content	KLDA acc.	under KLDA	under RanPAC
CIFAR-10	objects	96.94	0.04	0.00
CIFAR-100	objects	84.64	0.10	0.05
TinyImageNet	objects	79.90	0.08	−0.01
SUN397	scenes	73.55	0.06	−0.01
Stanford Cars	car models	66.69	0.01	−0.05
DTD	textures	69.20	0.07	0.05
RESISC45	aerial imagery	90.00	0.32	0.26
EuroSAT	satellite imagery	96.07	0.55	0.48
GTSRB	traffic signs	72.28	0.32	0.26
SVHN	street digits	59.05	0.35	0.33
MNIST	handwritten digits	98.23	0.49	0.38
first group, mean over 24 pairs			0.06	0.01
second group, mean over 20 pairs			0.41	0.34

K TEXT BENCHMARKS

Table 10: Four class-incremental text benchmarks, frozen BART-base, $D = 5000$, five-member ensemble, three seeds. One fixed configuration for FIRE (all 14 states, linear weights, $\gamma = 1$) and one fixed $\gamma = 10^{-4}$ for KLDA. The last column counts the seeds on which an exact paired McNemar test, run separately per seed, gives $p < 0.05$.

Dataset	Joint (pub.)	KLDA-E (pub.)	KLDA (ours)	FIRE	Δ	seeds $p<0.05$
HWU64	88.60	89.78	89.75	**90.74**	$+0.99$	1/3
Banking77	91.36	93.03	93.03	**93.80**	$+0.77$	0/3
CLINC150	95.33	96.62	96.22	**96.71**	$+0.49$	0/3
DBpedia	94.83	94.53	**94.43**	94.40	-0.03	0/3
Mean	92.53	93.49	93.36	**93.91**	$+0.55$	—

Our KLDA reproduction matches the published numbers to within 0.06 points on every dataset when each is given its own best bandwidth; the column in Table 10 is lower on two datasets because one bandwidth is fixed for all four. FIRE gains 0.55 points on average and wins on three datasets, but under per-seed tests only one seed on HWU64 reaches $p < 0.05$, so the text gains are not statistically established. Tuning both methods per dataset gives wins on all four and a mean of $+0.67$. The gains on text are smaller than those on the image benchmarks (§4.7). Within BART, decoder states beat encoder states on every dataset, and both halves together beat either alone except on DBpedia (Appendix W).

L PAIRED SIGNIFICANCE TESTS

ImageNet overlap. The two ViTs are trained with ImageNet labels; TinyImageNet is a 200-class subset of ImageNet-1k, and CIFAR-100 and Stanford Cars overlap partially. These two backbones also show the largest gains, $+5.14$ and $+4.51$. Comparisons within a column are unaffected, since both readouts use the same backbone; the effect on averages across backbones is given in §4.3.

Table 11: Paired tests over all 44 backbone–dataset pairs. A pair counts as significant only if an exact McNemar test (McNemar, 1947) clears $p < 0.05$ on *each* of the three random-feature seeds separately; predictions of the same test image under different seeds are correlated, so seeds are never pooled into one contingency table. The last column applies Holm–Bonferroni correction across the 44 pairs within each comparison, controlling the family-wise error rate without assuming the pairs are independent; it is applied to each pair's maximum per-seed p-value and counts significant pairs in either direction, with the gain sign required to agree across seeds. The maximum is a valid conservative p-value for the null that at least one seed has no effect; a raw median need not be. Intervals use 10,000 hierarchical bootstrap draws: sample eleven datasets with replacement, sample three seed indices with replacement within each sampled dataset, and average over the fixed panel of four backbones. These intervals characterize variation within this benchmark panel, not uncertainty over all possible backbones or unseen domains. Mean gains are computed from the per-image prediction dumps and differ from the table-based means quoted in the text (3.57, 2.66, 2.04) by at most 0.03 points.

Comparison	mean gain [95% CI]	wins	losses	sig. after Holm
Firing pattern vs. final layer, both under KLDA	$+3.55$ $[+1.49, +6.34]$	**36**	1	33
RanPAC vs. KLDA, identical final-layer features	$+2.05$ $[+0.92, +3.34]$	25	1	24
Firing pattern vs. final layer, both under RanPAC	$+2.63$ $[+0.66, +5.35]$	27	3	24

M PER-DATASET RESULTS ON DINOV2

Table 12: Per dataset on DINOv2-B/14, validation protocol, ordered by the gain the firing pattern gives the KLDA head. The two frozen heads and the joint linear probe on each readout; joint fine-tuning of the backbone; and the fraction of the distance from KLDA to fine-tuning that KLDA over the pattern closes. The two negative KLDA entries across all four backbones are CIFAR-10 here and Stanford Cars on CLIP (-2.57).

	final layer			firing pattern					
Dataset	KLDA	RanPAC	probe	KLDA	RanPAC	probe	FP−KLDA	fine-tune	closed
SVHN	59.64	61.10	61.77	74.42	75.00	79.44	+14.79	98.10	0.38
GTSRB	71.80	71.95	72.83	83.07	81.43	79.13	+11.27	99.11	0.41
RESISC45	91.64	91.69	91.45	93.99	93.76	94.10	+2.35	97.22	0.42
DTD	75.14	79.89	79.68	77.45	80.68	81.12	+2.31	81.54	0.36
SUN397	74.29	77.06	76.94	76.42	76.92	77.20	+2.13	78.26	0.54
EuroSAT	96.89	96.80	97.04	98.52	98.18	98.52	+1.63	99.11	0.73
TinyImageNet	82.79	88.05	88.63	83.95	87.24	88.69	+1.16	90.38	0.15
CIFAR-100	89.14	91.23	90.96	89.91	91.28	91.74	+0.77	93.88	0.16
MNIST	98.73	98.30	98.41	99.43	99.21	99.22	+0.70	99.75	0.69
Stanford Cars	85.77	87.11	87.25	86.31	86.78	87.36	+0.53	93.68	0.07
CIFAR-10	98.67	98.76	98.78	98.62	98.67	98.63	−0.05	99.10	−0.11
Mean	84.05	85.63	85.79	87.46	88.10	88.65	+3.42	93.65	0.35

N SPARSIFICATION AND SCHEDULE TABLES

Both on DINOv2 over eleven datasets with per-cell test-selected γ; the baselines therefore differ slightly from Table 2. Expert weighting is in Appendix P.

Table 13: Sparsification, $D = 5000$. τ is the fraction of each block's units kept, chosen by a per-class Fisher score or by per-image magnitude; memory is total classifier state, against 111.9 MB for the baseline.

τ	Fisher	per-image	vs. KLDA	mem. (MB)
0.05	79.81	—	−4.32	106.7
0.10	82.96	—	−1.17	116.4
0.25	85.55	83.79	+1.42	144.9
0.50	86.40	85.90	+2.28	192.5
1.00	**87.32**	—	+3.19	287.7

Table 14: Layer schedules, $\gamma = 0.4$.

Schedule	Mean	beats unif.
quadratic decreasing	59.11	0/11
linear decreasing	81.32	3/11
uniform	**87.29**	—
linear increasing	86.80	3/11
quadratic increasing	86.45	3/11
Fisher, measured	87.00	4/11

O MEMORY ACCOUNTING

Accuracy against total memory on DINOv2 over the ten datasets other than DTD (which is capped at $D \leq 800$), per-cell test-selected γ. *Classifier* is $\mathbf{\Omega}$, $\mathbf{\Sigma}$ and the per-class means in float32 on a 100-class benchmark; *backbone* is the frozen encoder, which must be resident to produce features. The backbone dominates until D exceeds roughly 3000, so a classifier-only count would report $0.38\times$ for the second row instead of $0.84\times$. Two resident encoders put a 659 MB floor under every fused configuration.

Configuration	D	classifier	backbones	total	accuracy
KLDA	5000	112 MB	330 MB	442 MB	84.94
FIRE	1000	42 MB	330 MB	**373 MB**	85.27
FIRE	2000	92 MB	330 MB	423 MB	87.02
FIRE	5000	288 MB	330 MB	618 MB	88.32
fusion, final layer	1000	10 MB	659 MB	671 MB	88.56
fusion, all layers	1000	75 MB	659 MB	735 MB	88.71
fusion, all layers	2000	157 MB	659 MB	817 MB	89.94
fusion, all layers	5000	449 MB	659 MB	1110 MB	**90.80**

P EXPERT BANK RESULTS PER TASK

Each expert evaluated on its own task with the task identity given, against the fused pattern choosing among all 758 classes without it. Means are weighted by test-set size.

Task	expert alone	fused, 758-way	Δ	test n
MNIST	99.74	99.75	$+0.01$	10,000
GTSRB	99.05	97.25	-1.80	12,630
EuroSAT	98.70	98.41	-0.29	2,700
SVHN	97.60	97.56	-0.04	26,032
RESISC45	92.48	90.18	-2.30	9,450
Stanford Cars	57.03	57.21	$+0.18$	8,041
SUN397	40.52	39.97	-0.55	15,880
DTD	27.39	**30.16**	$+2.77$	1,880
Weighted mean	81.78	81.22	-0.56	86,613

Weighting over experts	accuracy	task-hit
uniform	**81.22**	94.88
$\propto$ in-task accuracy	81.10	94.68
$\propto$ training-set size	80.85	95.13
$\propto$ class count	80.13	91.44

The bandwidth has little effect here: accuracy moves from 81.08 to 81.22 across a sixteenfold range of γ, far flatter than on any pretrained backbone.

Q STRICTLY CONTINUAL PROTOCOL, PER DATASET

Average incremental accuracy under the protocol of §4.6, averaged over the four backbones; $\pm$ is the standard deviation over five class orders, averaged over backbones. T is the number of tasks. RanPAC and ridge re-select their ridge at each step on the incoming task's validation rows only, which may favor plasticity over stability. On CIFAR-10 and MNIST the first task has two classes, validation accuracy ties at 100% across KLDA's bandwidth grid, the smallest bandwidth wins, and some sequences lose many points mid-way under either readout; this is the source of the larger spreads for those datasets.

		KLDA		RanPAC	
Dataset	T	final layer	firing pattern	final layer	firing pattern
CIFAR-10	5	98.24 ± 0.31	97.88 ± 1.02	98.29 ± 0.31	98.29 ± 0.32
CIFAR-100	10	89.60 ± 0.77	90.37 ± 0.69	90.91 ± 0.80	91.82 ± 0.71
TinyImageNet	10	85.41 ± 0.40	86.28 ± 0.34	87.99 ± 0.33	88.29 ± 0.31
Stanford Cars	10	71.13 ± 4.48	72.08 ± 0.92	78.37 ± 0.81	79.46 ± 0.72
DTD	5	79.00 ± 0.97	80.22 ± 0.89	83.89 ± 1.05	85.33 ± 0.87
SUN397	10	80.88 ± 0.47	82.19 ± 0.45	83.21 ± 0.42	83.51 ± 0.45
SVHN	5	68.53 ± 3.65	79.79 ± 3.45	69.38 ± 3.35	81.58 ± 2.67
GTSRB	9	80.76 ± 2.62	85.74 ± 2.19	80.25 ± 2.43	86.55 ± 1.98
MNIST	5	97.42 ± 2.69	97.72 ± 3.40	98.98 ± 0.16	99.51 ± 0.09
RESISC45	9	91.70 ± 0.57	93.65 ± 0.63	93.87 ± 0.64	95.84 ± 0.51
EuroSAT	5	97.76 ± 0.25	98.80 ± 0.15	97.79 ± 0.24	98.97 ± 0.14

RIDGE TRAJECTORIES UNDER THE STRICT PROTOCOL

Accuracy on all classes seen so far after each task, averaged over five class orders and four backbones, for ridge on the raw final block and on all thirteen raw blocks. The gap widens along the sequence on every high-gain dataset.

dataset	input	accuracy after task t									
SVHN	final	88.8	76.3	65.8	62.5	57.6					
	all 13	93.3	88.0	83.4	82.6	80.8					
GTSRB	final	91.9	88.7	85.3	81.8	79.6	77.5	75.1	72.8	70.9	
	all 13	93.8	92.5	91.0	90.3	89.4	88.2	86.8	86.1	85.4	
MNIST	final	99.8	99.3	98.6	97.9	97.5					
	all 13	99.9	99.7	99.6	99.5	99.4					
RESISC45	final	98.3	96.1	94.7	93.5	92.1	90.9	89.8	88.8	87.9	
	all 13	98.6	97.5	97.1	96.5	95.9	95.4	95.1	94.7	94.2	
EuroSAT	final	99.6	99.0	97.5	96.4	95.5					
	all 13	99.8	99.6	98.9	98.8	98.6					
CIFAR-100	final	97.1	94.5	93.0	91.2	89.8	88.3	87.1	86.1	85.0	84.2
	all 13	97.5	95.6	94.5	93.3	92.4	91.3	90.5	89.9	89.0	88.0
Stanford Cars	final	89.5	84.7	80.4	77.7	74.8	73.7	71.7	70.1	68.8	67.6
	all 13	91.4	87.3	84.2	82.4	79.5	78.7	78.0	77.0	75.9	74.3
SUN397	final	93.2	90.2	86.8	84.0	81.5	79.1	77.1	75.1	73.2	71.3
	all 13	93.5	91.0	87.7	85.4	83.0	81.3	79.7	78.1	76.6	75.5

R PER-PAIR UNCERTAINTY FOR THE MATCHED COMPARISONS

Paired differences between two configurations on the same backbone–dataset pair (matched features, Table 1), in accuracy points. Intervals are a hierarchical bootstrap of the mean (10,000 draws, resampling datasets and then backbones within each resampled dataset); wins and losses count pairs.

comparison	group	mean [95% CI]	wins	losses
normalization, ridge (all 13)	obj./scene	$+0.13$ [-0.06, $+0.40$]	12	12
	high-gain	-0.01 [-0.11, $+0.06$]	12	8
normalization, RanPAC (all 13)	obj./scene	-0.13 [-0.48, $+0.22$]	8	16
	high-gain	$+1.69$ [$+0.46$, $+3.37$]	17	3
normalization, KLDA (all 13)	obj./scene	-0.47 [-2.81, $+0.67$]	17	7
	high-gain	$+0.90$ [$+0.18$, $+2.16$]	18	2
ridge, all 13 vs. last 6 (raw)	obj./scene	$+0.25$ [$+0.07$, $+0.47$]	19	3
	high-gain	$+2.01$ [$+0.35$, $+4.43$]	19	1
ridge all 13 vs. RanPAC pattern	obj./scene	$+0.21$ [-0.53, $+1.21$]	10	14
	high-gain	$+2.88$ [$+0.38$, $+5.71$]	20	0
ridge all 13 vs. KLDA pattern	obj./scene	$+3.10$ [$+0.95$, $+6.89$]	22	2
	high-gain	$+2.84$ [$+0.61$, $+5.41$]	20	0
pattern vs. final, RanPAC	obj./scene	$+0.79$ [-0.06, $+1.84$]	17	6
	high-gain	$+5.85$ [$+1.77$, $+10.88$]	20	0
pattern vs. final, KLDA	obj./scene	$+0.57$ [-0.94, $+1.48$]	18	6
	high-gain	$+6.74$ [$+2.56$, $+11.47$]	20	0

S HELD-OUT DATASETS AND ADVANCE PREDICTIONS

The prediction was written down before any feature extraction or evaluation on these datasets. It classifies each dataset by image content: not natural photographs (characters, clothing silhouettes) predicts the high-gain group; natural photographs predicts the other. The criteria, fixed in advance, were: (1) under KLDA, every predicted-high dataset removes a larger fraction of baseline errors than every predicted-natural dataset; (2) predicted-high datasets remove at least 0.15 and predicted-natural datasets at most 0.12, the bounds of the two original groups; (3) criterion (1) also holds under RanPAC. One amendment was made before evaluation, after staging revealed the dataset sizes: KLDA's D is capped at half the training rows on STL-10 (2500), Pets (1800) and Flowers-102 (1000), the rule of the strictly continual protocol, to avoid the covariance-rank collapse of Appendix V. Data come from public mirrors (KMNIST, EMNIST-letters and STL-10 from the task-arithmetic suite, Fashion-MNIST, Oxford-IIIT Pets and Flowers-102 from their standard splits, with the Flowers validation images added to training). Accuracies are means over the four backbones; "removed" is the fraction of baseline errors removed by the pattern, averaged over backbones. The ridge columns were not part of the prediction. The three predicted-high datasets share the MNIST format; the boundary test below addresses that confound.

		KLDA			RanPAC			ridge (raw)	
dataset	predicted	final	pattern	removed	final	pattern	removed	final	all 13
KMNIST	high	86.11	92.35	+0.44	87.66	93.54	+0.48	78.78	95.64
EMNIST-letters	high	89.79	93.07	+0.31	89.91	93.25	+0.32	86.03	94.37
Fashion-MNIST	high	90.99	92.44	+0.16	91.37	92.86	+0.17	89.58	93.78
STL-10	natural	99.35	99.37	+0.03	99.44	99.42	−0.02	99.42	99.30
Oxford-IIIT Pets	natural	92.75	92.24	−0.06	93.52	92.23	−0.17	93.01	93.56
Flowers-102	natural	98.48	98.42	+0.07	99.11	98.98	−0.05	98.68	99.05

DATASET PROVENANCE

Held-out datasets: KMNIST (Clanuwat et al., 2018), EMNIST-letters (Cohen et al., 2017) and STL-10 (Coates et al., 2011) from the task-arithmetic mirrors (`tanganke/kmnist`, `tanganke/emnist_letters`, `tanganke/stl10`); Fashion-MNIST (Xiao et al., 2017) from `zalando-datasets/fashion_mnist`; Oxford-IIIT Pets (Parkhi et al., 2012) from `timm/oxford-iiit-pet`; Oxford Flowers-102 (Nilsback & Zisserman, 2008) from `dpdl-benchmark/oxford_flowers102`, with its validation images added to training. Boundary cases: MNIST-M (Ganin et al., 2016) from `Mike0307/MNIST-M` (59,001 training, 9,001 test images); USPS (Hull, 1994) from `flwrlabs/usps` (7,291 and 2,007). CIFAR-10-g28 and STL-10-g28 apply, before the backbone's own preprocessing, a bilinear resize to 28×28 and a conversion to single-channel grayscale; the backbone preprocessing then resizes to 224 pixels and replicates the channel. The second extraction of the original suite uses the task-arithmetic splits (Ilharco et al., 2023) through the same mirrors for Stanford Cars, DTD, SUN397, RESISC45 and EuroSAT.

Per-backbone results. The fractions of errors removed quoted in the text average over backbones, and near-ceiling datasets have few errors. *Held-out* columns give final-layer → all-block accuracy (KLDA and RanPAC: raw final to normalized all 13; ridge: raw final to raw all 13) and the number of test errors of the KLDA final-layer baseline. *Boundary* rows use the validation-selected best final embedding.

dataset	backbone	KLDA	RanPAC	ridge	base errors
KMNIST	DINOv2	86.91→93.45	88.83→94.33	79.47→96.06	1309
	CLIP	89.89→93.80	89.58→94.99	82.49→96.10	1011
	ViT-21k	83.04→91.19	85.46→92.57	75.95→95.27	1696
	ViT-1k	84.61→90.97	86.79→92.28	77.21→95.12	1539
EMNIST-l.	DINOv2	91.29→93.75	91.38→94.10	87.81→94.65	1812
	CLIP	92.39→94.07	92.16→94.15	89.57→94.77	1582
	ViT-21k	87.19→92.14	87.71→92.36	82.77→94.04	2665
	ViT-1k	88.28→92.30	88.39→92.40	83.98→94.01	2437
Fashion	DINOv2	92.16→93.06	92.52→93.43	90.89→94.24	784
	CLIP	90.33→92.08	90.59→92.50	89.02→93.24	967
	ViT-21k	90.55→92.22	91.00→92.67	89.05→93.84	945
	ViT-1k	90.93→92.39	91.38→92.85	89.37→93.78	907
STL-10	DINOv2	99.54→99.53	99.57→99.57	99.68→99.58	37
	CLIP	99.09→99.09	99.15→99.11	99.03→99.10	73
	ViT-21k	99.35→99.46	99.50→99.52	99.46→99.21	52
	ViT-1k	99.42→99.42	99.52→99.49	99.51→99.29	46
Pets	DINOv2	95.62→95.53	95.98→95.91	95.04→95.34	161
	CLIP	91.41→89.64	92.41→88.18	92.59→92.86	315
	ViT-21k	91.40→91.37	92.50→92.09	91.82→92.45	316
	ViT-1k	92.59→92.40	93.19→92.73	92.59→93.59	272
Flowers	DINOv2	99.62→99.66	99.68→99.70	99.69→99.61	24
	CLIP	96.77→96.02	98.05→97.48	96.57→97.84	199
	ViT-21k	98.95→99.26	99.35→99.38	99.27→99.41	65
	ViT-1k	98.59→98.75	99.39→99.37	99.19→99.35	87
MNIST-M	DINOv2	55.29→78.12	56.07→76.90	52.74→86.58	4024
	CLIP	88.16→92.67	88.55→92.77	86.76→94.18	1066
	ViT-21k	44.68→70.68	44.71→68.82	42.64→82.89	4980
	ViT-1k	49.60→72.38	49.78→70.95	46.83→83.62	4537
USPS	DINOv2	96.23→97.84	96.34→97.54	96.11→98.21	76
	CLIP	97.21→97.93	97.29→98.03	97.46→98.21	56
	ViT-21k	95.49→97.09	95.42→97.14	94.97→97.31	91
	ViT-1k	95.70→97.18	95.68→97.28	94.67→97.01	86
CIFAR-10-g28	DINOv2	90.80→91.44	90.99→91.82	90.05→92.19	920
	CLIP	84.79→86.55	85.20→87.11	83.19→87.86	1521
	ViT-21k	83.07→86.32	83.55→86.35	82.10→87.78	1693
	ViT-1k	84.43→87.07	85.02→87.16	83.17→88.37	1557
STL-10-g28	DINOv2	87.14→86.56	88.92→88.83	88.01→88.57	1029
	CLIP	77.65→77.84	79.45→80.42	78.87→80.51	1788
	ViT-21k	77.15→76.52	79.13→80.95	78.56→79.55	1828
	ViT-1k	77.92→77.89	80.24→81.25	79.80→81.02	1766

BOUNDARY TEST: CONTENT VERSUS FORMAT

Written before staging, feature extraction or evaluation of the four cases below. Metric: fraction of test errors removed by the all-block readout (raw all-13 for ridge, normalized all-13 for RanPAC and KLDA) relative to the best final-layer representation chosen per backbone, dataset and head on validation data among the raw last hidden state, the post-LayerNorm CLS state, CLIP's projected embedding and the ViT-21k pooler, each raw or normalized; mean over the four backbones. All runs use the extended search of Appendix T and the stable KLDA solve. Predictions (content hypothesis, for each head): (P1) MNIST-M and USPS each remove at least 0.15; (P2) the two converted cases each remove at most 0.12; (P3) every digit case removes more than every converted case; (P4) under ridge, all 13 blocks beat the validation-selected best single block on at least 3 of 4 backbones for MNIST-M and USPS. P1, P3 and P4 held under every head (P4: 4 of 4 for both); P2 failed under every head because of CIFAR-10.

case	content, format	ridge	RanPAC	KLDA
MNIST-M	digits, color photographic	$+0.67$	$+0.43$	$+0.45$
USPS	digits, 16×16 grayscale	$+0.44$	$+0.34$	$+0.35$
CIFAR-10-g28	natural, 28×28 grayscale	$+0.28$	$+0.13$	$+0.14$
STL-10-g28	natural, 28×28 grayscale	$+0.06$	$+0.04$	-0.02

T KLDA SOLVER ROBUSTNESS

Matched KLDA cells (Table 1) with the reference solve (unregularized float32 pseudoinverse) and with a float64 solve after shrinkage by 10^{-3} of the covariance's mean eigenvalue, fixed in advance and not tuned. Means over the same backbone–dataset pairs.

group	solve	final block, raw	all 13, raw	all 13, normalized
obj./scene	reference float32	79.53	80.57	80.09
	float64 + shrinkage	79.66	80.58	80.77
high-gain	reference float32	82.23	88.07	88.97
	float64 + shrinkage	83.57	88.16	89.20

With shrinkage, normalized KLDA on Stanford Cars with the ImageNet-1k ViT scores 58.9 against 54.5 raw, where the reference solve gave 40.4 against 55.1.

HYPERPARAMETER GRID EDGES

Number of matched cells (Table 1, seed 0) whose validation-selected hyperparameter lies on an edge of its grid: $\lambda \in \{10^{-8}, 10^{6}\}$ for ridge and RanPAC; $\gamma \in \{10^{-6}, 10^{-2}\}$ for raw and $\{0.05, 0.8\}$ for normalized KLDA inputs. An edge selection means the optimum may lie outside the grid.

head	raw final	raw all 13	normalized final	normalized all 13
ridge	22/44	6/44	14/44	8/44
RanPAC	28/44	14/44	3/44	0/44
KLDA	18/44	10/44	21/44	16/44

The 160 edge cells were re-run with an extended search: KLDA starts from $1/\text{median}\|x - x'\|^2$ on training rows and uses a grid spaced by factors of four for both raw and normalized inputs; ridge and RanPAC start from the decade grid; every grid is extended in steps of its own spacing until the optimum is interior, then refined locally; KLDA uses the float64 solve with shrinkage. After the search no selection lies on an edge. The remaining KLDA cells (raw final block, raw and normalized all 13) were then searched on the original grid followed by local refinement to about $\sqrt{2}$ resolution for both input types, so every KLDA cell below has comparable tuning. Matched means with those cells replaced (KLDA: stable solve throughout):

head	group	final, raw	all 13, raw	all 13, normalized
ridge	obj./scene	79.75	83.20	83.33
	high-gain	82.25	91.82	91.81
RanPAC	obj./scene	82.37	83.10	82.98
	high-gain	83.67	87.27	88.94
KLDA	obj./scene	80.41	81.38	81.37
	high-gain	83.64	88.61	89.22

Edge selections in the decisive controls (single-block sweep, last four blocks, all thirteen blocks and random views; cells counted over the two dataset groups) and in the strictly continual protocol (one selection per class order for KLDA, one per step for ridge and RanPAC). None of these runs used the extended search.

	ridge	RanPAC	KLDA
single block, last 4, all 13, random views	1/132	0/176	80/176
strict protocol, final layer	109/1660	1033/1660	95/220
strict protocol, all blocks	14/1660	9/1660	114/220

U NUMERICAL PRECISION

Numerics. TF32 matrix multiplication is disabled. KLDA uses a float32 pseudoinverse without covariance shrinkage. The matched ridge and RanPAC comparisons accumulate and solve their second-order systems in float64; Appendix X distinguishes these from the original grid. With TF32 enabled, one identical command produced accuracies ten points apart on three GPU models, as detailed below.

An early image sweep enabled TF32 matrix multiplication for speed. The identical command, same seed and same cached features, produced 0.7224 on a V100 (which has no TF32 path), 0.6759 on

an A100 and 0.6219 on an A40: deterministic within each GPU model and ten points apart across them. With TF32 disabled the three agree to 0.7223, 0.7224 and 0.7228.

The discrepancy comes from the conditioning of $\mathbf{\Sigma}$. TF32 keeps ten mantissa bits, a relative rounding error near 4.9×10^{-4} against 6×10^{-8} for float32, and inverting $\mathbf{\Sigma}$ amplifies it by roughly $\mathrm{cond}(\mathbf{\Sigma})$. On SVHN, $\mathrm{cond}(\mathbf{\Sigma}) = 6.3 \times 10^4$, so the first-order conditioning bound is already larger than one. This does not measure the actual inverse error, but it explains why precision requires an empirical check: Mahalanobis scoring amplifies errors in small-eigenvalue directions. All contaminated results were discarded and re-run in float32 with TF32 off, and the feature cache was re-extracted at full precision, since float16 storage had been a second confound at a comparable mantissa width. A large seed variance we had earlier observed on SVHN was three seeds landing on three GPU models.

V Rank of the Shared Covariance

$\mathbf{\Sigma}$ is estimated from $n - |\mathcal{C}|$ residuals, so its rank is at most $n - |\mathcal{C}|$. The pseudoinverse discards directions below its numerical cutoff, so the damage comes from the other end of the spectrum: as D approaches the bound the smallest retained singular values shrink towards the numerical threshold and are estimated from almost no data, and inverting them amplifies noise until accuracy collapses to chance with no error raised. DTD has 1,880 training images over 47 classes, or forty samples per class. At $D = 1832$, the scatter is nominally full rank but poorly conditioned, and test accuracy falls to 2.2% for both methods. Accuracy recovers to 75.1 for KLDA and 77.4 for FIRE at $D = 800$ and declines again by $D = 1000$, so D has an interior optimum. The behavior is inherited unchanged from KLDA and never arises at the scale of the original text experiments ($n = 10^4$, $D = 5000$).

W Layer Subsets on BART

Splitting BART-base into its encoder and decoder halves, each variant at its own best γ. Decoder states beat encoder states on all four datasets, and the full pattern is best everywhere except DBpedia, where the decoder half alone leads. The two halves contribute the same number of dimensions and differ only in where they sit along the stack.

Dataset	final layer only	encoder (7)	decoder (7)	all (14)
CLINC150	96.53	96.76	96.93	**97.02**
Banking77	93.00	93.27	93.47	**93.80**
DBpedia	94.50	93.40	**94.63**	94.10
HWU64	89.31	89.62	90.80	**90.95**

X Implementation Details and Evaluation Provenance

Checkpoints and preprocessing. The image checkpoints are `facebook/dinov2-base`, `openai/clip-vit-base-patch16`, `google/vit-base-patch16-224-in21k` and `google/vit-base-patch16-224`; the text checkpoint is `facebook/bart-base`. The extraction script resizes with bicubic interpolation, center-crops to the processor's declared crop size (224 for these image runs), converts to RGB and applies the processor's channel means and standard deviations. It caches the CLS token of every returned hidden state in float32. The final-layer baseline is the last of these states, before the backbone's final normalization or projection, rather than the model's separately exported pooled embedding. Checkpoint revisions (Hugging Face commit hashes, first eight characters): DINOv2-B/14 `f9e44c81`, CLIP ViT-B/16 `57c21647`, ViT-B/16 IN-21k `b4569560`, ViT-B/16 IN-1k `3f49326e`. The exact 10% selection holdout of every dataset is recorded in a split manifest to be released with the code.

Matched comparison and layer controls. The training holdout is stratified with fraction 0.1 and split seed 0. The matched ridge and RanPAC grid is $\lambda \in \{10^{-8}, 10^{-7}, \ldots, 10^{6}\}$. KLDA uses $\gamma \in \{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$ for raw inputs and $\{0.05, 0.1, 0.2, 0.4, 0.8\}$ for normalized inputs. The matched script selects with random-feature seed 0, then refits and evaluates seeds $0, 1, 2$. Ridge has no random projection, so these repeated fits are not independent evidence. KLDA also

caps D below $n_{\text{fit}} - |\mathcal{C}|$ in the comparison scripts; DTD's earlier $D = 800$ choice remains test-informed. Layer selection and bandwidth selection use the same validation split. RanPAC layer controls choose the largest ridge within 0.1 percentage points of the best validation accuracy; the matched script uses the first maximizer in its ordered grid. The KLDA bandwidth grid for raw features spans decades (10^{-6} to 10^{-2}), while the grid for normalized features is spaced by factors of two (0.05 to 0.8); cells whose selection fell on an edge were re-searched with the extended grid described in Appendix T.

Original grid. The original grid selects shared hyperparameters per backbone and readout. In DINOv2, CLIP, IN-21k, IN-1k order, the saved KLDA final-layer bandwidths are $(1.5{\times}10^{-4}, 7.392{\times}10^{-3}, 6.882\times10^{-6}, 6.921\times10^{-6})$, and the all-layer bandwidths are $(0.2, 0.4, 0.2, 0.2)$. RanPAC's final-layer ridges are $(10^6, 10^4, 10^7, 10^7)$, and its normalized all-layer ridge is 10^2 for all four. This original grid predates the float64 ridge solve used in the matched comparisons; the matched table should therefore be used for conclusions about the ranking of heads.

Solvers. KLDA forms the unnormalized within-class scatter in float32 and calls `torch.pinverse` without an explicit tolerance override or shrinkage term. The PyTorch default relative cutoff for that API is 10^{-15}. The small cutoff is a limitation in float32 near the covariance rank, not a regularization guarantee. Matched ridge and RanPAC accumulate their Gram and feature–label cross-product matrices in float64 and solve the ridge system in float64, returning classifier weights in the input dtype. TF32 is disabled. A streaming implementation should update counts, sums and outer-product sums; calling a complete-class scatter update on arbitrary pieces of the same class would require an additional mean-correction term.

Incremental selection. Five class orders use order seeds 0 through 4 and random-feature seed 0. KLDA fixes its map from task one, with $D \le \lfloor n_1/2 \rfloor$ and bandwidth selected from the corresponding raw or normalized grid above. Its first-maximizer tie rule selects the smallest bandwidth when all validation accuracies tie. RanPAC keeps its random projection fixed and reselects ridge using only current-task validation rows and accumulated training statistics; near-ties favor the largest ridge as above. The reported evaluation reconstructs each prefix from cached features of seen classes, rather than benchmarking an online accumulator's memory or runtime. Direct ridge is evaluated under the same protocol, reselecting its ridge per step exactly as for RanPAC but without the random projection.

Joint references. The linear probe selects a penalty from $\{10^{-6}, \ldots, 10^{-1}\}$ and either per-coordinate standardization or a shared RMS scale, with a learned intercept and at most 300 L-BFGS iterations. Scaling statistics are computed from the training set, including the training holdout, but never the test set. Joint fine-tuning uses seed 0, ten epochs of AdamW, batch size 64, backbone learning rate 2×10^{-5}, head learning rate 10^{-3} and weight decay 0.05. It uses one warmup epoch followed by cosine decay, label smoothing 0.1, gradient-norm clipping at 1, and automatic mixed precision. This single recipe is a reference point, not an optimized upper bound.

Evaluation protocols. Each table's numbers come from one of four protocols.

protocol	tables	selection and solver
original grid	2, 3, 11, 12	original extraction; one hyperparameter per backbone and readout, chosen on a 10% holdout; reference KLDA solve
matched	1, 4 (top)	second extraction; per-cell selection on a 10% holdout over fixed grids; float64 ridge and RanPAC, reference KLDA solve
matched, extended search	grid-edge appendix; boundary test	as matched, with scale-aware grids extended until interior and refined; float64 KLDA with shrinkage
strictly continual	4 (bottom), per-dataset incremental appendix	KLDA bandwidth and D from the first task; ridge and RanPAC re-selected per step on incoming-task holdout rows

Runtime and state. Median fit time for one final fit on the full training set and deployed classifier state over the matched high-gain and three object/scene cells, on a single GPU of mixed type (RTX A6000, L40S or A100): ridge on the final block 0.1 s and 0.1 MB; ridge on all thirteen raw blocks

9.1 s and 1.8 MB of weights, plus the $9{,}984^2$ Gram matrix during accumulation (about 800 MB in float64); RanPAC 9.4 s and 33 MB on the final block, 10.0 s and 401 MB on all blocks; KLDA 7.3 s and 116 MB, and 7.8 s and 300 MB. Peak device memory was not recorded. Timings exclude feature extraction, which is shared by all readouts.

Limits of reconstruction. The released code and split manifest will specify the second extraction completely. Per-example splits of the original extraction cannot be regenerated, because that feature cache is no longer available; the tables built from it (Tables 2, 3 and the original-grid appendices) are therefore reproducible only up to the splits described in the text. Earlier test-informed choices and the change in extraction prevent treating the whole study as one evaluation planned in advance.

Y NUMBER OF RUNS

Roughly 6,100 evaluation runs, feature extraction on four frozen image backbones and one text backbone, and eight from-scratch expert trainings, over 15 datasets with 3 seeds. The largest batches are the clean image grid (1,386 runs covering the baseline, the dense pattern and both sparsification rules at four fractions), the bandwidth sweep across all four backbones (792), the D sweep behind Appendix F (648), the per-backbone baseline grids (495), the schedule sweep (198), the text grid (240), the paired-prediction runs (132 image, 48 text), backbone fusion and its D sweep (570), the three- and four-backbone fusion (528), expert fusion (120), the joint linear probes (88) and the joint fine-tuning runs (11). An earlier image round of 594 runs was discarded as TF32-contaminated (Appendix U) and is not reused anywhere.