

Fisher information and the geometry of memorization in neural networks

Daniel T. Bernstein*
Princeton University¹

Chase Goddard*
Princeton University²

Luca Di Carlo
Princeton University^{1,2}

Vudtiwat Ngampruetikorn§
University of Sydney³

David J. Schwab§
CUNY Graduate Center⁴

Abstract

A fruitful approach towards understanding generalization in machine learning involves characterizing how models separate generalizable structure from idiosyncratic or even corrupted features of training data. Fisher information has emerged as a diagnostic tool for this separation, leveraging loss geometry to distinguish generalization from memorization. Here we investigate this connection in vision models trained on CIFAR-10 with controlled noise that can be learned only through memorization. We show that projecting model weights onto the top Fisher eigenvectors decouples general task performance from noisy sample memorization, and we investigate how this decoupling presents differentially across layers. We show that the same phenomena are present in small transformers trained to solve a synthetic in-context learning task, and are reproduced in closed form in a simple, analytically solvable linear model. Our results reveal that Fisher information captures aspects of the geometric structure underlying how neural networks allocate capacity between generalization and memorization.

1 Introduction

Modern neural networks can memorize their training set [1], yet in practice they learn features that generalize to unseen examples. Disentangling memorization from generalization is a central problem in machine learning—large language models reproduce substantial fractions of their pretraining data [2], raising copyright and privacy concerns; adversaries can exploit memorization through data poisoning [3]; and memorization of mislabeled examples degrades generalization. Progress on any of these fronts requires a clearer picture of how trained models store memorized content and what distinguishes it from generalizing features.

The geometry of the loss landscape offers a natural setting for this question, and recent work argues that the eigenspectrum of the Fisher information separates memorized information from generalizing features, using this separation to remove memorized content from trained language models [4]. Here, we investigate this phenomenon across several settings designed to probe how and when the Fisher encodes memorized features.

The Fisher information’s averaging structure suggests why it might isolate memorization. On a dataset D , the Fisher decomposes as an average of per-sample contributions, $F = \frac{1}{|D|} \sum_{x \in D} F_x$, where each F_x measures the sensitivity of the network’s output to its parameters at sample x . Directions that encode generalizing features should align across single-sample Fishers and add coherently in this

*DTB and CG contributed to this work equally.

§VN and DJS contributed to this work equally.

¹Lewis-Sigler Institute ²Joseph Henry Laboratories ³School of Physics ⁴Initiative for the Theoretical Sciences

average. On the other hand, directions that encode memorization of particular samples should instead be largely orthogonal in parameter space and suppressed by a factor $\sim 1/|D|$ in the average. The top of the population Fisher’s eigenspectrum is therefore a natural place to look for a separation between memorized content and generalizing structure.

This picture turns memorization into a set of testable geometric predictions. If memorized features are supported on sample-specific directions, then projecting onto the top Fisher eigenspaces should preserve structure shared across the training distribution while suppressing information tied to individual examples. In a trained network, however, the relevant eigenspaces may depend on the data used to compute the Fisher, may appear only in particular layers, and may trade off memorization removal against the loss of weak but useful features. These considerations motivate our experiments. The Gaussian-image setting isolates pure input-level memorization (§3.1), the label-noise setting tests whether de-memorization can recover an underlying correct label (§3.2) and the in-context regression setting asks whether the same geometry applies to memorized latent tasks (§3.3). The barcode model then distills the same mechanism into a linear setting where finite-sample correlations with uninformative features can be analyzed in closed form (§5).

Our work extends this Fisher-based account of memorization in three directions. First, we show that distinct forms of memorized content—random-input examples, mislabeled training examples, and memorized latent tasks in a synthetic in-context learning task—leave distinct signatures in the layer-wise Fisher geometry. Second, we find that the geometric separation between memorization and generalization depends sharply on the data used to evaluate the Fisher: a denoised dataset yields cleaner separation than the full training distribution. Third, suppressing memorized directions can partially recover task generalization that the trained model otherwise loses to overfitting.

1.1 Contributions

In this work, we study the relationship between sample memorization and the Fisher information. We do so by investigating conditions under which the projection of a neural network’s weights onto a subspace governed by the Fisher information induces de-memorization. We define de-memorization as an operation performed on a neural network which disproportionately degrades its performance on samples that the network was forced to memorize, while inducing only minor degradation on other samples. We emphasize that de-memorization is separate from the phenomenon of unlearning: de-memorization does not allow a network to selectively forget individual samples in its training set [5–7]. We specify the projection operation in detail in the Experiments section. Our contributions are as follows:

- We demonstrate how low-eigenvalue subspaces of the Fisher information encode memorized data across a variety of experimental settings: We investigate input/output noise in image classifiers, the structure of subspaces across layers, and an in-context regression task.
- We show that de-memorization performance depends on the samples used to compute the Fisher: the structure of Fisher eigenspaces with respect to memorization is data-dependent.
- In an in-context regression problem, we demonstrate that de-memorization can be lifted to the level of latent tasks while simultaneously improving task generalization, and we identify model components responsible for task memorization.
- We investigate how easily individual samples can be de-memorized from the model by using information encoded in single-sample Fisher matrices, and we use the language of participation ratios to understand how different kinds of samples contribute to different regions of the eigenspectrum.
- We develop an analytically tractable “barcode” model that exhibits the de-memorization phenomenon in a simple linear setting.

2 Related Work

Curvature-based signatures of memorization. Recent works have used local curvature measures to identify example-level memorization. Most closely related to our work, Merullo *et al.* [4] study the eigenspectrum of loss curvature in language models, finding that the top Fisher modes encode reasoning-like structure, whereas lower-curvature modes encode memorized samples. Loss curvature

per-sample has been proposed as a way to identify mislabeled and conflicting examples, and corrupted data injected into the training set can exhibit high-curvature signatures [8]. Influence-function methods use gradients and Hessian-vector products to map predictions back to influential training examples [9].

Memorization and noisy labels. Overparameterized neural networks can fit arbitrary labels and random inputs [1]. However, networks also tend to learn simple, “shared” structure before fitting noisy samples [10], which has inspired methods that can leverage this tendency to train networks on very noisy data [11]. Memorization can also be beneficial for generalization, and influence-based approaches suggest that memorization of atypical examples can improve performance on rare subpopulations [12, 13].

Loss geometry, sharpness, and Fisher information. A significant body of work relates generalization to the local geometry of the loss landscape. Flatter minima have long been hypothesized to generalize better [14], and empirical studies have connected sharp minima to generalization gaps [15, 16]. Studies of the Hessian spectrum have shown that trained networks typically exhibit a small set of stiff directions along with a bulk of near-zero eigenvalues [17–19]. The Fisher information provides a related local metric [20, 21] which does not rely on a loss function to measure model sensitivity. Furthermore, there is evidence that flat directions connect minima more globally [22–25]. Fisher information has been used to identify parameter directions important for previously learned tasks in continual learning [26] and to study early phase training dynamics [27].

In-context learning and task overfitting. Regression tasks have become a standard testbed for studying in-context learning [28–30]. More recent work shows that the emergence of in-context learning depends strongly on task diversity and training duration, such that transformers may first generalize to held-out tasks and later overfit by memorizing the finite set of training tasks [31–33]. Further work localizes the memorization to the MLP layers in the transformer [32, 33].

Benign overfitting and data coarse-graining. Our theoretical model studies a simplified setting of high-dimensional linear regression. In work on benign overfitting [34, 35], fitting pure noise can increase variance, and removing uninformative feature directions after fitting can reduce risk in finite samples. Our model isolates a “barcode” block that is uninformative across the population but is correlated with potential label noise. Data coarse-graining work shows that discarding data dimensions can improve predictive performance when the discarded degrees of freedom primarily encode noise [36].

Unlearning. The de-memorization phenomena we study is distinct from unlearning research. Unlearning refers to the procedure of forcing a neural network to forget specific information on which it was trained, and is usually studied in the context of data privacy concerns or data which might produce harmful behavior in the model [5–7]. The type of model editing that we investigate does not support fine-grained deletion of information required for unlearning.

3 Experiments

To explore the interplay between memorization and loss geometry, we force models to memorize a subset of their training data using two variants of the standard CIFAR-10 classification task [37].

First, to study input noise, we augment the CIFAR-10 training set with 500 synthetic images (1% of the training set). Following standard protocols for injected noise (see, e.g., Ref [1]), each synthetic image is drawn from a Gaussian distribution whose mean and variance match the statistics of a randomly chosen CIFAR-10 class, but is assigned the label of an independently chosen random class. Because the input statistics and labels are independent, the network cannot rely on generalizable features, forcing it to explicitly memorize these inputs.

Second, to study label noise, we randomly corrupt 10% of the training labels (see, e.g., Ref [8]). Each corrupted label is reassigned uniformly among the nine incorrect classes, forcing the network to explicitly memorize these modified samples.

In both settings, we train a ResNet-20 [38] to 100% training accuracy (details in Appendix A.1). Following Merullo *et al.* [4], we compute and diagonalize each layer’s Fisher information matrix and

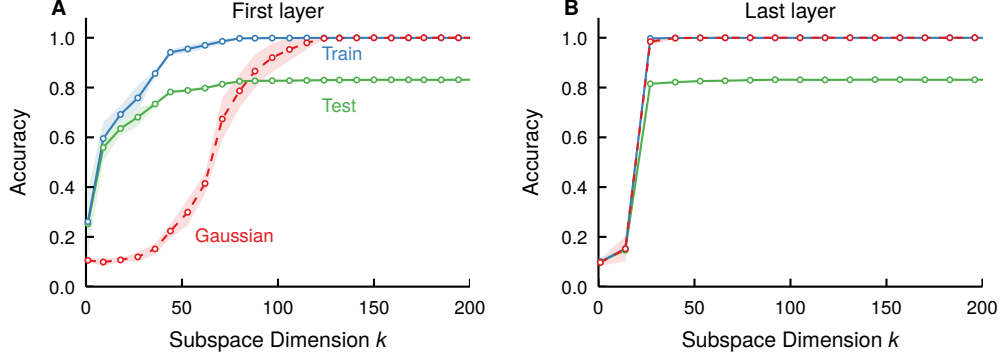


Figure 1: **Layer-dependent decoupling of memorization and generalization via Fisher projection: memorized Gaussian images.** Classification accuracy as a function of the number of retained top Fisher eigenvectors k , computed over the full training set (CIFAR-10 and Gaussian images) for a single layer of a ResNet-20. Green: CIFAR-10 test; blue: CIFAR-10 train; red: memorized Gaussian samples. **A)** In the first layer, CIFAR-10 train and test accuracy rise in tandem and reach a maximum at $k \approx 45$, while accuracy on memorized Gaussian samples remains at $\approx 30\%$, demonstrating selective de-memorization at negligible cost to generalization. **B)** In the last layer, all three curves rise sharply even for very small k , with no separation between them at any point, indicating that memorized and structured representations have merged and can no longer be separated.

evaluate performance after projecting weights onto the top- k subspaces. Unlike Ref [4], which relies on a K-FAC approximation, we compute the exact per-layer Fisher matrix. The Fisher information measures the sensitivity of the network’s output to parameter perturbations,

$$\widehat{\mathbf{F}}(\theta^*) = \frac{1}{N} \sum_{i=1}^N \sum_{c=1}^C p_{\theta^*}(c | \mathbf{x}_i) \mathbf{s}_{ic}(\theta^*) \mathbf{s}_{ic}(\theta^*)^\top, \quad \mathbf{s}_{ic}(\theta^*) \equiv \nabla_{\theta} \log p_{\theta}(c | \mathbf{x}_i)|_{\theta=\theta^*} \quad (1)$$

It is the Hessian of the Kullback-Leibler divergence, $D_{\text{KL}}(\theta || \theta') \approx \frac{1}{2}(\theta - \theta')^\top F(\theta)(\theta - \theta')$ for $\theta' \approx \theta$. We will show later that the dataset D over which we compute the expectation can significantly influence the results of this procedure.

3.1 Which layers store memorized information?

Figure 1 illustrates the effect of this procedure on the first and last layers of a network trained on the Gaussian data. We compute the Fisher information over the entire training set, including both CIFAR and Gaussian images.

In the first layer, the model’s accuracy on standard CIFAR-10 images (both train and test) rises quickly as a function of the eigenspace dimension k , outpacing its accuracy on the Gaussian data. At an eigenspace dimension of $k \approx 45$, the model achieves near-maximum accuracy on the CIFAR-10 dataset while correctly classifying less than 30% of the Gaussian samples. The accuracy on both the CIFAR-10 train and test sets rises in tandem, showing that this procedure separates the memorized data distribution from the learned distribution of CIFAR-10 images.

Applying this projection to the final layer results in markedly different behavior. Accuracy across all three partitions (CIFAR-10 train, CIFAR-10 test and Gaussian) climbs simultaneously to full-model accuracy as the subspace dimension k increases. By this layer, the model’s representations of the two data distributions have converged, precluding selective removal. Taken together with the first layer, this indicates that the signature of memorization is layer-dependent, restricting de-memorization interventions to specific layers.

We also note that nearly all the relevant predictive information in the final layer’s weights is concentrated within the span of the top $\sim 5\%$ of its eigendirections.

3.2 Memorization and generalization

In the Gaussian setting, de-memorization simply changes the model’s predictions on unstructured inputs. When the model has memorized mislabeled examples, however, a correct answer exists, and

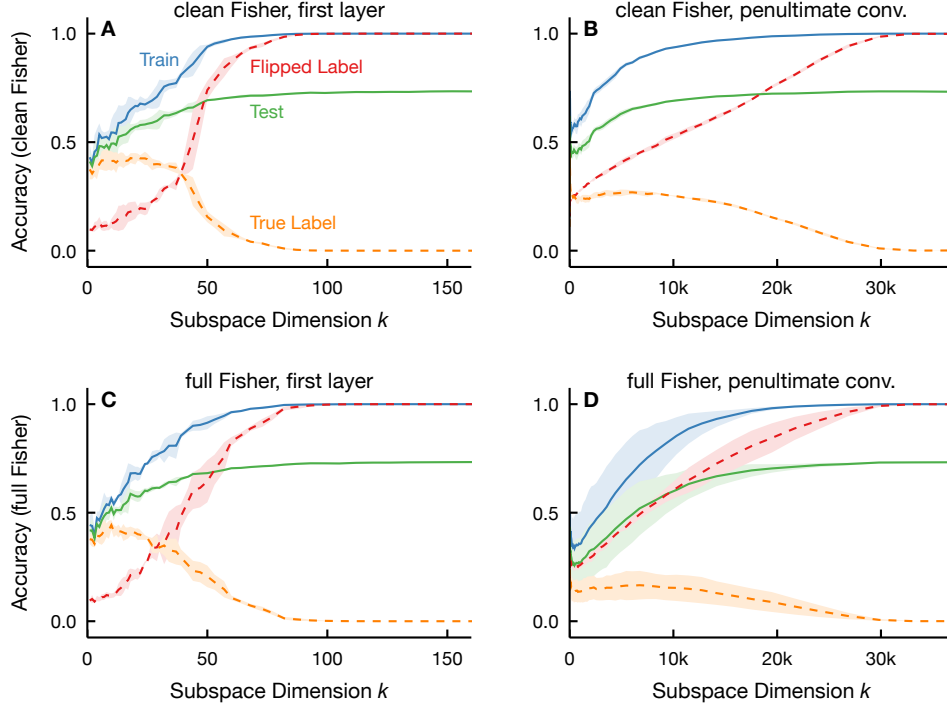


Figure 2: **Layer-dependent decoupling of memorization and generalization via Fisher projection: noisy labels.** Classification accuracy as a function of retained top Fisher eigenvectors k for a ResNet-20 trained on CIFAR-10 with 10% label noise. Blue: train accuracy; Green: test accuracy; Red: accuracy on poisoned samples under their (wrong) training labels; Orange: accuracy on poisoned samples under their true labels. **A&B)** Fisher computed on clean (non-corrupted) data only. **C&D)** Fisher computed on the full training set. **First layer, A&C):** top Fisher eigenspaces treat memorized and clean samples differently under both Fisher variants. **Penultimate convolutional layer, B):** with the clean Fisher, de-memorization is successful — the red curve has not reached its maximum by the time train and test accuracy do — and the model partially recovers the true labels of label-flipped samples, with the orange curve rising well above chance. **Penultimate convolutional layer, D):** with the full Fisher, decoupling becomes weaker, highlighting the role of dataset choice in computing the Fisher.

we can ask whether de-memorization recovers it. To answer this question, we turn to the second setting, where we flip the labels of 10% of the training set. When we “de-memorize” in this setting, we can ask how often the true label is recovered.

In Fig 2, we show the results of the projection procedure for two representative layers (for all other layers, see Appendix Figs 9 and 10). In the top row, we compute the Fisher information over a dataset of only clean examples and in the bottom row we compute the Fisher over a subset of the full training data. For the first layer (left column), we see similar behavior for both Fisher sets: the top subspace treats the data with real labels differently than the data with flipped labels. Although in this case the projection procedure does not completely remove the memorized data, we note that accuracy on the true labels of the flipped samples (orange curve) improves well above chance in the top Fisher subspace.

In the right column, we consider the penultimate convolutional layer, the third-to-last layer in the network. In this layer, the data on which the Fisher is computed matters significantly: when we use only clean samples, the top eigenspace again treats the two types of data differently, but when we use samples from the full training set, including those with noisy labels, the accuracy curves move closer together. This highlights a fundamental property of the Fisher information: its geometric structure is highly dependent on the data distribution over which it is evaluated.

What are the implications of the layer-wise analysis for the geometry of the full Fisher? Although computing the full Fisher is unwieldy for these models, we can use a block diagonal approximation

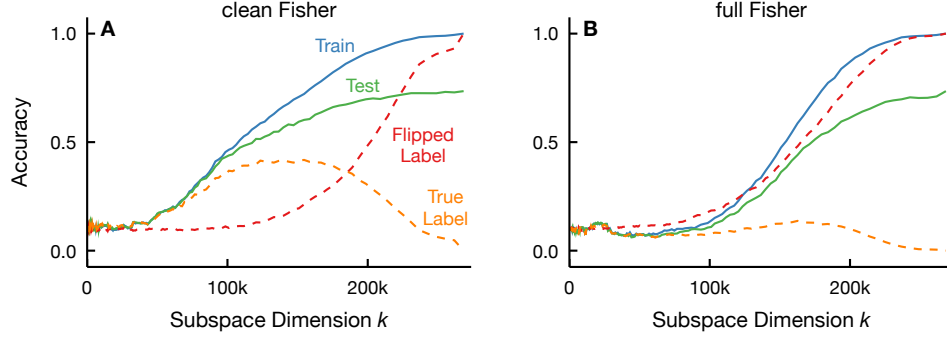


Figure 3: **Memorization and generalization across layers.** Results of using a block-diagonal approximation to estimate the Fisher information across all model parameters in the noisy-label setting. **A)** The Fisher is computed on training data with clean labels only. The top eigenspace selectively captures information about the true classification task. We also see non-monotonic behavior in the performance on the true labels of memorized samples (orange dashed line). **B)** The Fisher is computed on a sample from the full training dataset, including samples with corrupted labels. In this case, top eigenspaces mix information across tasks, emphasizing the data-dependence of the Fisher.

to gain some insight into how layer-wise phenomena accumulate across the model. In Fig 3, we show the results of the projection scheme for a block diagonal Fisher computed on clean data only (A) and a sample of the full training set (B).

When computing the Fisher matrix exclusively on data with clean labels (Fig 3A), the top eigenspaces selectively capture information about the true classification task. In these dominant subspaces, the network starts to ignore the corrupted labels it was forced to memorize during training. However, when we compute the Fisher on a sample from the full training dataset (Fig 3B), this geometric separation breaks down. Without the clean data filtering, the top eigenspaces mix information across the generalization and memorization tasks—in order to see any separability, we need to turn to the individual layers (Fig 2).

3.3 In-context learning

We next investigate memorization in an in-context learning (ICL) setting [39]. A growing body of work has developed in-context learning of linear regression as a simple model of ICL amenable to quantitative study [28, 31, 40, 41]. One important feature of ICL captured by the linear regression model is that ICL behavior can be *transient*, emerging early on during training before the model memorizes the set of training tasks [31–33]. Here, we investigate how the model encodes these memorized tasks, and show that we can recover at least some of the transient ICL performance using Fisher projection.

We train a three-block GPT-2 style transformer on contexts $C_\tau = (x_1, y_1, x_2, y_2, \dots, x_n)$ with $y_i = w_\tau^T x_i$. The model must learn to effectively infer the latent regression “task” vector w_τ to predict y_n . Following Ref [31], training contexts are sampled uniformly from a set of M task vectors, $\{w_\tau\}_{\tau=1}^M$. As shown in Fig 4A, when M is small, performance on held-out tasks first improves until the model overfits to the training tasks. We emphasize that the model never directly sees the task vectors, and so this setting, we investigate memorization of *latent* “concepts” rather than the direct memorization of the input data.

We next compute the Fisher information for each component in the model, and project the model parameters onto top- k subspaces. In Fig 4B and C, we show the results for the second transformer block’s attention and MLP layers (See Appendix Fig 8 for all layers). We see that in the attention layer, projecting onto top subspace only hurts model performance, on both the memorized training tasks and held-out test tasks. However, the same projection procedure applied to the MLP layer yields a substantial improvement in test performance at the cost of performance on the memorized tasks. We note that prior work also identified the MLP as the component of a transformer responsible for memorization [32, 33]; our analysis further suggests that this memorized information is stored in low-Fisher-eigenvalue parameter components. Moreover, removing memorized information can

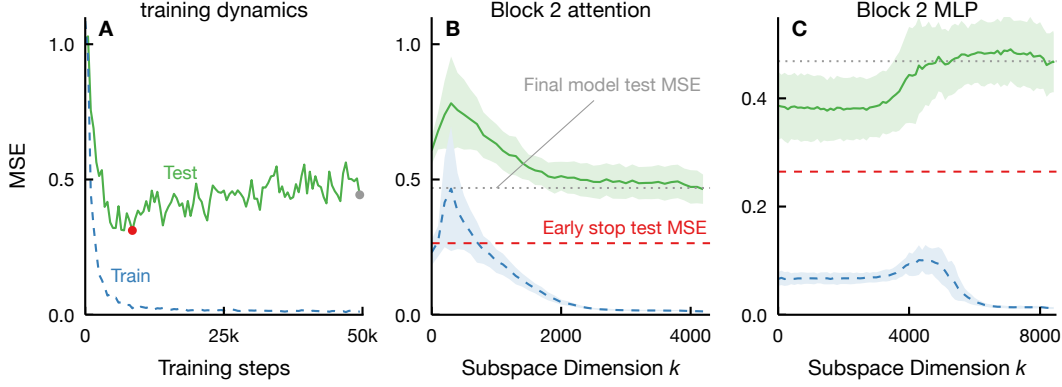


Figure 4: **From memorization to generalization, in-context.** **A)** Transient ICL training dynamics. The three-block transformer first improves performance on held-out tasks, before overfitting to training tasks. **B)** Fisher projections for the attention layer, second transformer block. All projections decrease performance. Shaded region shows standard deviation across 16 models. **C)** Fisher projections for the MLP layer, second transformer block. Top Fisher subspaces improve task generalization (green) at the cost of performance on memorized tasks (blue).

partially restore in-context learning based task generalization, closing roughly half the gap to the best early-stopped performance.

4 Contributions of Individual Samples

4.1 Sample Contributions to the Eigenspectrum

To better understand how individual data samples contribute to generalization and the structure of the Fisher information, we examine the participation ratios associated with individual samples in the eigenspectrum of the (layerwise) Fisher. To get some intuition, we can express the Fisher information as an average over per-sample Fishers: $\frac{1}{n} \sum_{i=1}^n F_i$. Then, diagonalizing the Fisher as $F = \sum_{k=1}^P \lambda_k \vec{u}_k \vec{u}_k^\top$, we define the contribution of sample i to mode k as $c_i^k = \vec{u}_k^\top F_i \vec{u}_k$. The participation ratio for mode k is then $\Psi^k = (\sum_i c_i^k)^2 / (n \sum_i (c_i^k)^2)$. This ratio is a measure of the fraction of data samples that contribute to mode k . For example, when $c_i^k = 1$ for all i , $\Psi^k = 1$ and all samples contribute (equally). On the other hand, when only a single sample contributes, $\Psi^k = \frac{1}{n}$ and the participation ratio is suppressed by the size of the dataset. The restricted participation ratio $\Psi_{\text{CIFAR}}^k \equiv (\sum_{j \in \text{CIFAR}} c_j^k)^2 / (n \sum_i (c_i^k)^2)$ restricts the numerator to only the contributions of the CIFAR samples, and this convention is used equivalently for $\Psi_{\text{Gauss}}^k, \Psi_{\text{corrupted}}^k$, and $\Psi_{\text{uncorrupted}}^k$. Note that $\Psi^k \neq \Psi_{\text{CIFAR}}^k + \Psi_{\text{Gauss}}^k$ generically because of cross-terms.

Figure 5A and B display first-layer participation ratios across the eigenspectrum. In the input-noise experiments (Fig 5A), we see total participation ratios drop by roughly one order of magnitude from top to bottom modes. Importantly, the restricted ratios show that clean CIFAR-10 images dominate the top modes, whereas memorized Gaussian noise is relegated to the bottom. These findings suggest that the top eigenmodes capture information about structured, generalizing features in the data, while the bottom portion corresponds to memorized, idiosyncratic information. By contrast, in the noisy-label experiments (Fig 5B), we see that corrupted and uncorrupted samples contribute in a constant ratio across the entire spectrum.

4.2 Single-Sample Projections

We also investigate single-sample deletion by projecting out top eigenvectors of single-sample Fisher information matrices, as we show in Fig 5C and D. We refer to the process of calculating a sample’s single-sample Fisher information and projecting out top eigenvectors of this Fisher as ‘sample projection.’

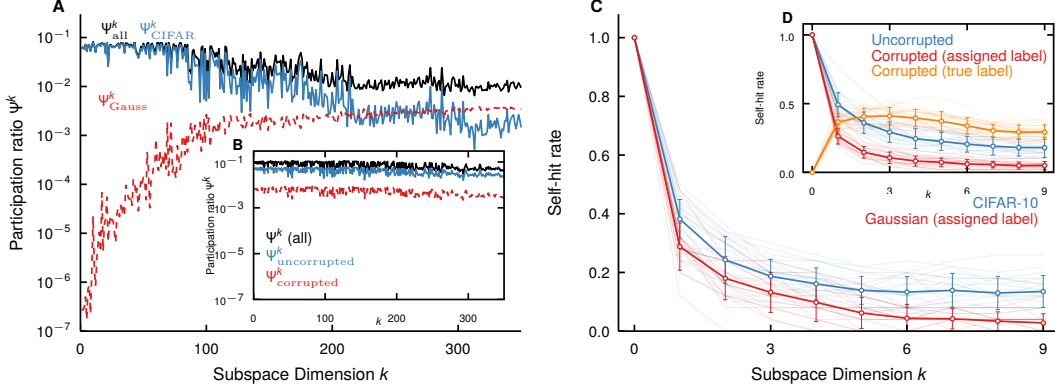


Figure 5: **Single-sample contributions and projections.** Participation ratios for **A)** input-noise case and **B)** label-noise case. **A)** suggests that structured CIFAR images primarily contribute to the top of the eigenspectrum, while noisy samples primarily contribute to the bottom of the eigenspectrum. This separation does not occur in **B)**. **C)** We plot the self-hit rate of correct sample classification after projecting out the top- k subspace of the corresponding single-sample Fisher information (‘sample projection’) in the input-noise case, and **D)** shows the same statistic for the label-noise case. Noisy samples are less likely to be recovered than their counterparts.

Table 1: Bulk accuracy after sample projection. The $k = 0$ column is the matched unchanged-weight BatchNorm control (see Appendix D for results on other Fisher sources).

Gaussian model						Noisy-label model (10% noise)					
	$k = 0$	1	2	4	9		$k = 0$	1	2	4	9
<i>Fisher source CIFAR-10 (%)</i>											
CIFAR-10 train (5k samples)	100.0	99.9	99.9	99.9	99.8	CIFAR-10 train (5k samples)	100.0	100.0	100.0	99.9	99.9
CIFAR-10 test	84.3	83.9	83.7	83.4	83.1	CIFAR-10 test	73.4	73.1	73.0	72.7	72.3
<i>Fisher source Gaussian (%)</i>											
Gaussian set	100.0	98.4	96.1	90.5	75.6	Corrupted CIFAR (assigned labels)	100.0	100.0	100.0	99.9	99.8

As depicted in Fig 5, classification accuracy after projection is higher for clean CIFAR-10 images than for Gaussian inputs, and this gap widens as more eigenvectors are projected out. Likewise, in the label-noise experiments, not only are uncorrupted samples more resilient to projection, corrupted samples also recover their true labels more often than their assigned false labels. True-class recovery is non-monotonic in the number of removed subspace dimensions. Finally, we note that sample projections minimally degrade samples besides the one targeted: the exception to this trend is significant degradation of performance on the Gaussian set after projecting out a Gaussian sample (see Table 1).

5 A barcode model of label-noise memorization

Our experiments demonstrate that projecting model weights onto top Fisher modes degrades the model’s recall of memorized samples while preserving generalization to clean data. Curvature-based editing [4] and coarse-graining of uninformative features [34, 36] offer partial accounts of this effect, but closed-form analyses for discrete label flips remain unavailable. To illustrate this mechanism, we develop an idealized linear model that yields exact expressions for the risk reduction from projection and reveals a regime in which an interior-optimal projection rank emerges.

Fig 6A depicts the generative model. For each input-label pair (x, y) , we generate a sign vector $u \sim \text{Uniform}(\{-1, +1\}^r)$. The feature vector x consists of informative features s , obtained from anisotropically scaling the sign vector, and barcode features z , modeled as an isotropic Gaussian vector; that is,

$$x = \begin{bmatrix} s \\ z \end{bmatrix} \in \mathbb{R}^{r+d} \quad \text{with} \quad s_i = \mu_i u_i \quad \text{and} \quad z \sim \mathcal{N}(0, I_d), \quad (2)$$

where μ is a fixed nonnegative vector. The clean label y is a scaled sign vector, and the corrupted training label $\tilde{y}$ is subject to random per-dimension sign flips,

$$y_i = a_i u_i \quad \text{and} \quad \tilde{y}_i = \eta_i y_i \quad (3)$$

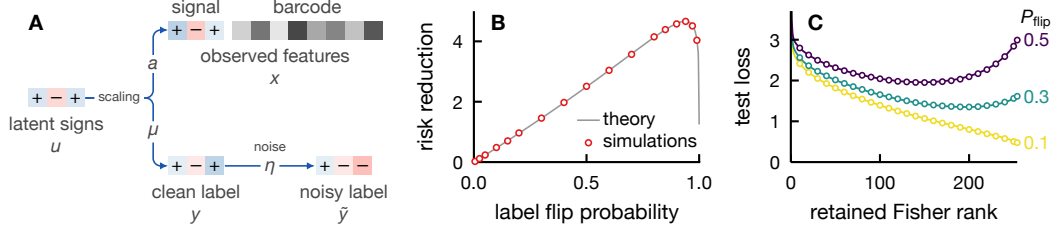


Figure 6: Finite-rank linear model illustrates the risk gain from Fisher projection. **A)** Generative model. A latent sign vector u is drawn uniformly from $\{-1, +1\}^r$; anisotropic scaling by amplitude vectors μ and a produces signal features s and clean label y , respectively. The observed features x result from concatenating the signal s and the uninformative barcode features z , modeled as an isotropic Gaussian vector in d dimensions. The training label $\tilde{y}$ is the clean label y with random per-dimension sign flips, with a random sign vector η . **B)** Risk gain from removing the barcode block post hoc vs label flip probability P_{flip} , demonstrating quantitative agreement between theory (solid curve, Eq (7)) and simulations (circles; error bars are smaller than the circle size). **C)** Prediction MSE vs retained Fisher rank for three flip rates (see legend). Two of four signal dimensions have population variance below the spectral separation threshold and therefore mix with the barcode bulk. In both theoretical prediction (solid curve, Eq (22)) and simulations (circles), an interior optimum emerges as the flip rate increases. Shown results are for $(n, r, d) = (500, 4, 250)$ with 200 replicates.

where a denotes a fixed nonnegative vector, $\eta \in \{-1, +1\}^r$ a random vector with $P(\eta_i = -1) = \rho_i$. The probability that at least one coordinate flips is $P_{\text{flip}} = 1 - \prod_i (1 - \rho_i)$. The corrupted label decomposes into a shrunk clean label and a noise term,

$$\tilde{y}_i = \bar{\eta}_i y_i + \xi_i, \quad (4)$$

where $\bar{\eta}_i \equiv \mathbb{E}[\eta_i] = 1 - 2\rho_i$. The noise term is given by

$$\xi_i = (\eta_i - \bar{\eta}_i) y_i \quad \text{with} \quad \mathbb{E}[\xi_j | y] = 0 \quad \text{and} \quad \mathbb{E}[\xi_i \xi_j | y] = \delta_{ij} a_i^2 (1 - \bar{\eta}_i^2). \quad (5)$$

Stacking n independent training samples yields the design matrix $X = \begin{bmatrix} S \\ Z \end{bmatrix} \in \mathbb{R}^{(r+d) \times n}$, with informative-feature block $S \in \mathbb{R}^{r \times n}$, barcode block $Z \in \mathbb{R}^{d \times n}$, corrupted-label matrix $\tilde{Y} \in \mathbb{R}^{r \times n}$ and noise matrix $\Xi \in \mathbb{R}^{r \times n}$.

We fit a linear model, $\hat{y} = \hat{\Theta}_s s + \hat{\Theta}_z z$, on the training data $(X, \tilde{Y})$ and evaluate its prediction risk on a fresh sample (x, y) from the *clean* generative model, i.e., no corrupted label at test time.

Minimizing $\|\tilde{Y} - \Theta_s S - \Theta_z Z\|_F^2$ over coefficient blocks $\Theta_s \in \mathbb{R}^{r \times r}$ and $\Theta_z \in \mathbb{R}^{r \times d}$ gives

$$\hat{\Theta}_z = \Xi \Pi_S^\perp Z^\top (Z \Pi_S^\perp Z^\top)^{-1}, \quad (6)$$

where $\Pi_S^\perp = I_n - S^\top (SS^\top)^{-1} S$ projects onto the complement of the informative-feature row space. This coefficient captures finite-sample correlation between the barcode design Z and the centered label noise Ξ . It vanishes in the infinite-data limit since z and ξ are uncorrelated.

We now calculate the effect of a post-hoc deletion of the barcode block. To do this, we consider the model $\hat{y} = \hat{\Theta}_s s$ instead of $\hat{y} = \hat{\Theta}_s s + \hat{\Theta}_z z$.¹ This intervention reduces prediction risk by

$$\Delta R = \mathbb{E} \|y - \hat{\Theta}_s s - \hat{\Theta}_z z\|^2 - \mathbb{E} \|y - \hat{\Theta}_s s\|^2 = \text{Tr}(\Sigma_\xi) \frac{d}{n - r - d - 1}, \quad \text{for } n - r > d + 1, \quad (7)$$

where the expectation is over both the training set $(X, \tilde{Y})$, on which the estimators depend, and a fresh test point (x, y) . This expression is equivalent to the variance that a d -dimensional linear regression with Gaussian covariates accumulates when fitting pure noise with $n - r$ effective samples. The condition $n - r > d + 1$ requires that the barcode dimension of the local model sit below the effective sample count; it does not constrain the total parameter count of a larger model. In Fig 6B, we depict the reduction in prediction risk, resulting from barcode deletion, over the full range of label

¹This procedure is not the same as deleting the barcode before fitting. Using both S and Z allows predictive and barcode features to compete, as both are correlated with the corrupted training labels in the finite-sample regime.

flip probability P_{flip} . We observe good agreement between the theoretical prediction, Eq (7), and numerical simulations.

Crucially, equation (7) assumes a clean separation between the informative and barcode directions, but Fisher projection on a real network operates in the empirical eigenmodes of the local feature covariance, where the two can mix. When an informative feature has population variance μ_j^2 below the spectral separation threshold $1 + \sqrt{d/(n-r)}$, its empirical eigenvector merges into the barcode bulk [42]. In this case, discarding low-eigenvalue directions sacrifices predictive information as well as memorized noise. Whether a finite-rank intervention benefits then depends on the regime. At low flip rates, the prediction-risk curve is monotone in retained rank and full rank is best. At higher flip rates, on the other hand, an interior-optimal rank emerges, set by the trade-off between memorization variance and informative-feature leakage. Fig 6C depicts this transition in the linear model; Appendix C.3 develops the partial-overlap theory underlying the solid curves.

6 Discussion and Outlook

Across a range of settings, our work shows that the memorized and generalized information within a neural network is separated by its Fisher geometry. In particular, retaining only the top of the eigenspectrum of the Fisher in certain layers induces de-memorization. Our findings suggest a picture of memorization which is both diffuse and localizable: single-layer interventions allow us to de-memorize noisy samples without degrading task performance on structured data, but we also observe an accumulation of layerwise phenomena in the Fisher information of the entire model.

Prior work has shown that adversarial competition exists between memorizing and generalizing circuits when language models are trained to perform in-context learning [33, 43]. Our work suggests that, in some cases, de-memorization can improve generalization without the need for retraining. Interference between the model’s memorizing and generalizing components seem to “hide” its full generalization capacity: by projecting out a certain fraction of components, we improve performance on tasks which benefit from generalization.

At the scale of the entire network, we find that the Fisher-geometric signature of memorization exhibits a strong data-dependence: it is easier to disentangle memorization from generalization when the Fisher is computed on a denoised version of the training data. The collapse of this disentanglability occurs in particular layers of the network, rather than within each layer.

We remark on two surprising features of the above phenomena from the perspective of information geometry. First, we have related memorization to a linear decomposition of parameter space. Second, we identify memorizing/generalizing subspaces by means of a local geometric quantity (the Fisher), but the projection procedure we perform is a non-local transformation: zeroing out eigenvectors can cause large perturbations to the network’s parameters. Prior work on loss-landscape geometry suggests that, in high-dimensional optimization problems, many soft directions exist in which large parameter-space perturbations correspond to small function-space perturbations, forming a so-called hyperribbon [44]. One possibility is that the nonlocal projection operation in parameter space corresponds to a small perturbation in the space of models. We leave an investigation of this possibility to future work.

Our results should be interpreted as an exploration of the geometric structure of memorization and generalization, not as a scale-independent prescription for model editing. Our empirical analysis uses ResNet-20 classifiers on CIFAR-10 variants and small transformers trained on synthetic in-context regression. The layerwise separation we observe may change qualitatively in larger models. Our vision experiments focus on simple object classification with controlled inputs and label noise; other data modalities may induce different forms of memorization—particularly when the memorized content is semantic, compositional or distributed across examples. We also emphasize that our analysis ignores cross-layer correlations, which may contribute meaningfully to memorization.

Our work contributes to a growing body of research which supports the possibility of “intelligent yet ignorant” models: models which can reason, but which have no knowledge of particular information. In particular, we have provided a number of examples where learning a task and memorizing data can be decoupled, and we demonstrate a geometric mechanism for this decoupling.

Acknowledgments and Disclosure of Funding

VN was supported in part by grants from the NSF (DMS-2235451) and Simons Foundation (MPS-NITMB-00005320) to the NSF-Simons National Institute for Theory and Mathematics in Biology (NITMB). DJS was partially supported by a Simons Fellowship in the MMLS, a Sloan Fellowship, and the National Science Foundation, through the Center for the Physics of Biological Function (PHY-1734030). VN and DJS also acknowledge funding from Coefficient Giving.

References

- [1] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, Understanding Deep Learning Requires Rethinking Generalization, in *International Conference on Learning Representations* (2017).
- [2] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, *et al.*, Extracting training data from large language models, in *30th USENIX Security Symposium (USENIX Security 21)* (2021) pp. 2633–2650.
- [3] M. Goldblum, D. Tsipras, C. Xie, X. Chen, A. Schwarzschild, D. Song, A. Madry, B. Li, and T. Goldstein, Dataset Security for Machine Learning: Data Poisoning, Backdoor Attacks, and Defenses, *CoRR* [abs/2012.10544](#) (2020), [2012.10544](#).
- [4] J. Merullo, S. Vatsavaya, L. Bushnaq, and O. Lewis, From Memorization to Reasoning in the Spectrum of Loss Curvature (2025), [arXiv:2510.24256 \[cs.CL\]](#).
- [5] Y. Cao and J. Yang, Towards Making Systems Forget with Machine Unlearning, in *2015 IEEE Symposium on Security and Privacy* (IEEE, 2015) pp. 463–480.
- [6] A. Ginart, M. Y. Guan, G. Valiant, and J. Zou, Making AI Forget You: Data Deletion in Machine Learning, in *Advances in Neural Information Processing Systems*, Vol. 32 (Curran Associates, Inc., 2019) pp. 3513–3526.
- [7] C. Guo, T. Goldstein, A. Hannun, and L. van der Maaten, Certified Data Removal from Machine Learning Models, in *Proceedings of the 37th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 119 (PMLR, 2020) pp. 3832–3842.
- [8] I. Garg, D. Ravikumar, and K. Roy, Memorization through the Lens of Curvature of Loss Function around Samples, in *Proceedings of the 41st International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 235 (PMLR, 2024).
- [9] P. W. Koh and P. Liang, Understanding Black-box Predictions via Influence Functions, in *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 70 (PMLR, 2017) pp. 1885–1894.
- [10] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, and S. Lacoste-Julien, A Closer Look at Memorization in Deep Networks, in *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 70 (PMLR, 2017) pp. 233–242.
- [11] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, Co-teaching: Robust Training of Deep Neural Networks with Extremely Noisy Labels, in *Advances in Neural Information Processing Systems*, Vol. 31 (2018) pp. 8527–8537.
- [12] V. Feldman, Does Learning Require Memorization? A Short Tale about a Long Tail, in *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing* (ACM, 2020) pp. 954–959.
- [13] V. Feldman and C. Zhang, What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation, in *Advances in Neural Information Processing Systems*, Vol. 33 (2020) pp. 2881–2891.
- [14] S. Hochreiter and J. Schmidhuber, Flat Minima, *Neural Computation* **9**, 1 (1997).
- [15] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima, in *International Conference on Learning Representations* (2017).

- [16] Y. Jiang, B. Neyshabur, H. Mobahi, D. Krishnan, and S. Bengio, Fantastic Generalization Measures and Where to Find Them, in *International Conference on Learning Representations* (2020).
- [17] L. Sagun, L. Bottou, and Y. LeCun, Eigenvalues of the Hessian in Deep Learning: Singularity and Beyond, in *International Conference on Learning Representations Workshop* (2017).
- [18] V. Pappas, The Full Spectrum of Deepnet Hessians at Scale: Dynamics with Sample Size (2018), [arXiv:1811.07062 \[stat.ML\]](#).
- [19] B. Ghorbani, S. Krishnan, and Y. Xiao, An Investigation into Neural Net Optimization via Hessian Eigenvalue Density, in *Proceedings of the 36th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 97 (PMLR, 2019) pp. 2232–2241.
- [20] S.-i. Amari, Natural Gradient Works Efficiently in Learning, *Neural Computation* **10**, 251 (1998).
- [21] J. Martens, New Insights and Perspectives on the Natural Gradient Method, *Journal of Machine Learning Research* **21**, 1 (2020).
- [22] T. Garipov, P. Izmailov, D. Podoprikin, D. P. Vetrov, and A. G. Wilson, Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs, in *Advances in Neural Information Processing Systems*, Vol. 31, edited by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Curran Associates, Inc., 2018).
- [23] F. Draxler, K. Veschgini, M. Salmhofer, and F. Hamprecht, Essentially No Barriers in Neural Network Energy Landscape, in *Proceedings of the 35th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 80, edited by J. Dy and A. Krause (PMLR, 2018) pp. 1309–1318.
- [24] J. Frankle, G. K. Dziugaite, D. M. Roy, and M. Carbin, Linear mode connectivity and the lottery ticket hypothesis, in *Proceedings of the 37th International Conference on Machine Learning*, ICML’20 (JMLR.org, 2020).
- [25] L. D. Carlo, C. Goddard, and D. J. Schwab, Entropic Confinement and Mode Connectivity in Overparameterized Neural Networks, in *The Fourteenth International Conference on Learning Representations* (2026).
- [26] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, Overcoming Catastrophic Forgetting in Neural Networks, *Proceedings of the National Academy of Sciences* **114**, 3521 (2017).
- [27] S. Jastrzebski, D. Arpit, O. Astrand, G. Kerg, H. Wang, C. Xiong, R. Socher, K. Cho, and K. J. Geras, Catastrophic Fisher Explosion: Early Phase Fisher Matrix Impacts Generalization, in *Proceedings of the 38th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 139 (PMLR, 2021) pp. 4772–4784.
- [28] S. Garg, D. Tsipras, P. Liang, and G. Valiant, *What Can Transformers Learn In-Context? A Case Study of Simple Function Classes* (2023), [arXiv:2208.01066 \[cs.CL\]](#).
- [29] E. Akyürek, D. Schuurmans, J. Andreas, T. Ma, and D. Zhou, What Learning Algorithm Is In-Context Learning? Investigations with Linear Models, in *International Conference on Learning Representations* (2023).
- [30] J. von Oswald, E. Niklasson, E. Randazzo, J. Sacramento, A. Mordvintsev, A. Zhmoginov, and M. Vladymyrov, Transformers Learn In-Context by Gradient Descent, in *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 202 (PMLR, 2023) pp. 35151–35174.
- [31] A. Raventós, M. Paul, F. Chen, and S. Ganguli, Pretraining task diversity and the emergence of non-Bayesian in-context learning for regression, in *Advances in Neural Information Processing Systems*, Vol. 36, edited by A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Curran Associates, Inc., 2023) pp. 14228–14246.
- [32] A. K. Singh, S. C. Chan, T. Moskovitz, E. Grant, A. M. Saxe, and F. Hill, The Transient Nature of Emergent In-Context Learning in Transformers, in *Thirty-seventh Conference on Neural Information Processing Systems* (2023).

- [33] A. Nguyen and G. Reddy, Differential learning kinetics govern the transition from memorization to generalization during in-context learning, in *The Thirteenth International Conference on Learning Representations* (2025).
- [34] P. L. Bartlett, P. M. Long, G. Lugosi, and A. Tsigler, Benign overfitting in linear regression, *Proceedings of the National Academy of Sciences* **117**, 30063 (2020).
- [35] T. Hastie, A. Montanari, S. Rosset, and R. J. Tibshirani, Surprises in High-Dimensional Ridgeless Least Squares Interpolation, *The Annals of Statistics* **50**, 949 (2022).
- [36] A. Nguyen, D. J. Schwab, and V. Ngampruetikorn, Data coarse graining can improve model performance, *Phys. Rev. E* **113**, 025301 (2026).
- [37] A. Krizhevsky, G. Hinton, *et al.*, Learning multiple layers of features from tiny images. (2009).
- [38] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016) pp. 770–778.
- [39] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, Language Models are Few-Shot Learners, in *Advances in Neural Information Processing Systems*, Vol. 33, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin (Curran Associates, Inc., 2020) pp. 1877–1901.
- [40] C. Goddard, L. M. Smith, V. Ngampruetikorn, and D. J. Schwab, When can in-context learning generalize out of task distribution?, in *Proceedings of the 42nd International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 267, edited by A. Singh, M. Fazel, D. Hsu, S. Lacoste-Julien, F. Berkenkamp, T. Maharaj, K. Wagstaff, and J. Zhu (PMLR, 2025) pp. 19585–19599.
- [41] J. Hill, B. Eyre, and E. Creager, *Transformers Don’t In-Context Learn Least Squares Regression* (2025), [arXiv:2507.09440 \[cs.LG\]](https://arxiv.org/abs/2507.09440).
- [42] J. Baik, G. B. Arous, and S. Péché, Phase Transition of the Largest Eigenvalue for Nonnull Complex Sample Covariance Matrices, *The Annals of Probability* **33**, 1643 (2005).
- [43] C. Gibson, W. Cui, and G. Reddy, *Distinct mechanisms underlying in-context learning in transformers* (2026), [arXiv:2604.12151 \[cs.LG\]](https://arxiv.org/abs/2604.12151).
- [44] K. N. Quinn, M. C. Abbott, M. K. Transtrum, B. B. Machta, and J. P. Sethna, Information geometry for multiparameter models: new perspectives on the origin of simplicity, *Reports on Progress in Physics* **86**, 035901 (2022).
- [45] T. Anderson, *An Introduction to Multivariate Statistical Analysis*, 3rd ed. (John Wiley & Sons, Hoboken, NJ, 2003).

A Additional Experimental Details

A.1 Model Training

We train all Resnet-20 models for 200 epochs with an initial learning rate of 0.1, and quench the learning rate by a factor 10 at epoch 100 and epoch 150. We use a batch size of 128 and train the models using SGD with weight decay 10^{-4} and momentum 0.9. Each model is trained on a single GPU. The model training requires 1-2 GPU hours, and the Fisher projection experiments require ~ 12 GPU hours per model on an 80GB GPU. We are careful to re-estimate batchnorm parameters after each projection.

The transformer used in section 3.3 is a standard GPT-2 style transformer with learned positional embeddings, hidden dimension 32, and consists of three transformer blocks and a linear embedding from the regression dimension to the hidden dimension. We use 8 attention heads in each block, and dropout 0.1. We generate sequences of 20 (x, y) pairs at training and test time, and $x \in \mathbb{R}^5$.

B Supplemental figures

In Fig 7 we show projections for all layers in the input-noise setting. In Fig 8, we show all transformer blocks for our ICL setting. In Fig 9 and Fig 10, we show projections for all layers in the label-noise setting, both for a clean Fisher and the Fisher of the full training set. In Fig 11 we show the results of control experiments in which we project models iteratively onto an alternative basis to the Fisher eigenbasis.

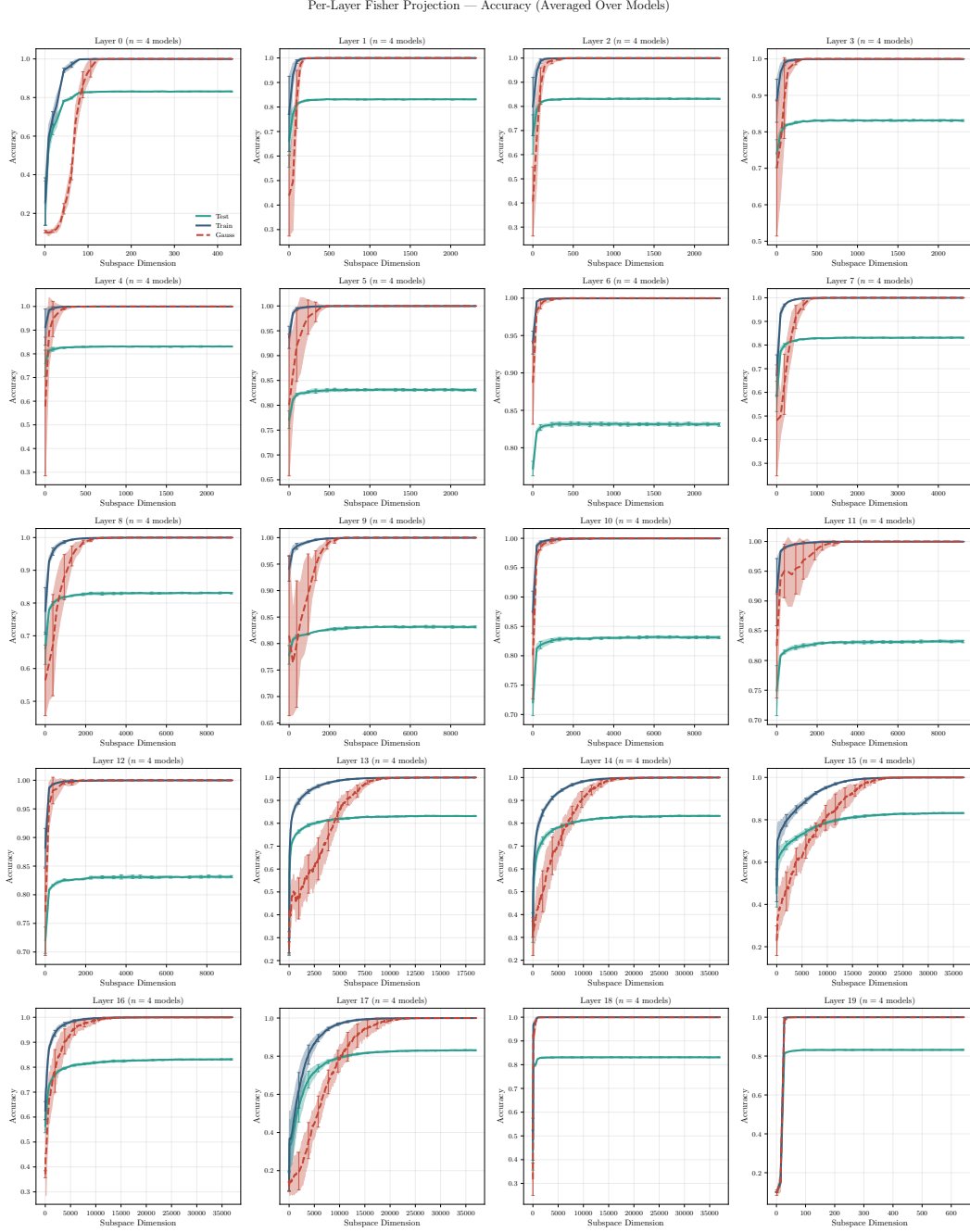


Figure 7: Layerwise Fisher projection experiments for the input-noise case.

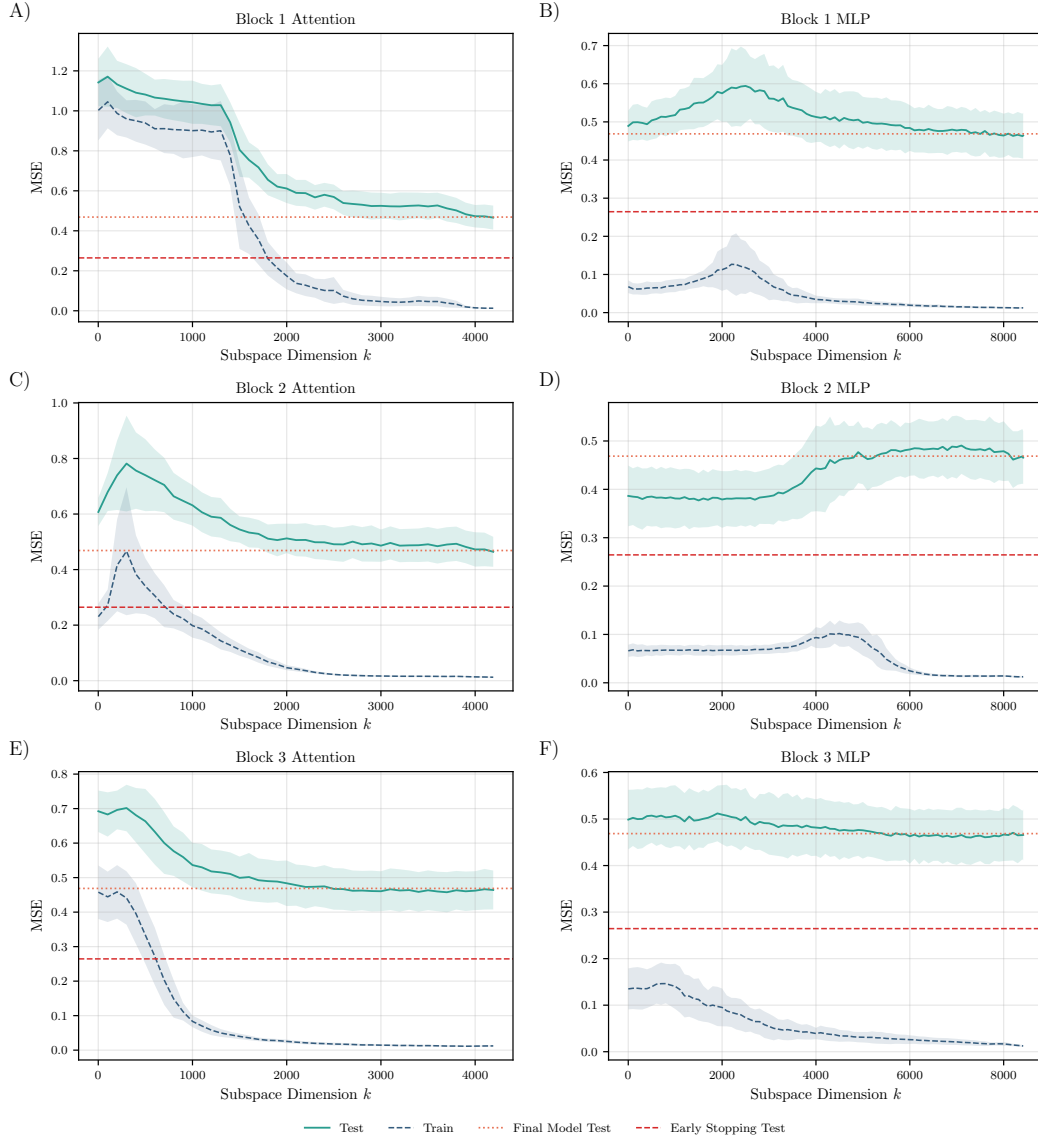


Figure 8: **Memorization concentrates in a single MLP.** Here, we show all transformer blocks for the model considered in Fig 4. All layers except the Block 2 MLP perform worse on both train/test tasks when restricted to a top Fisher subspace, but in the Block 2 MLP test performance improves at the cost of performance on the training tasks.

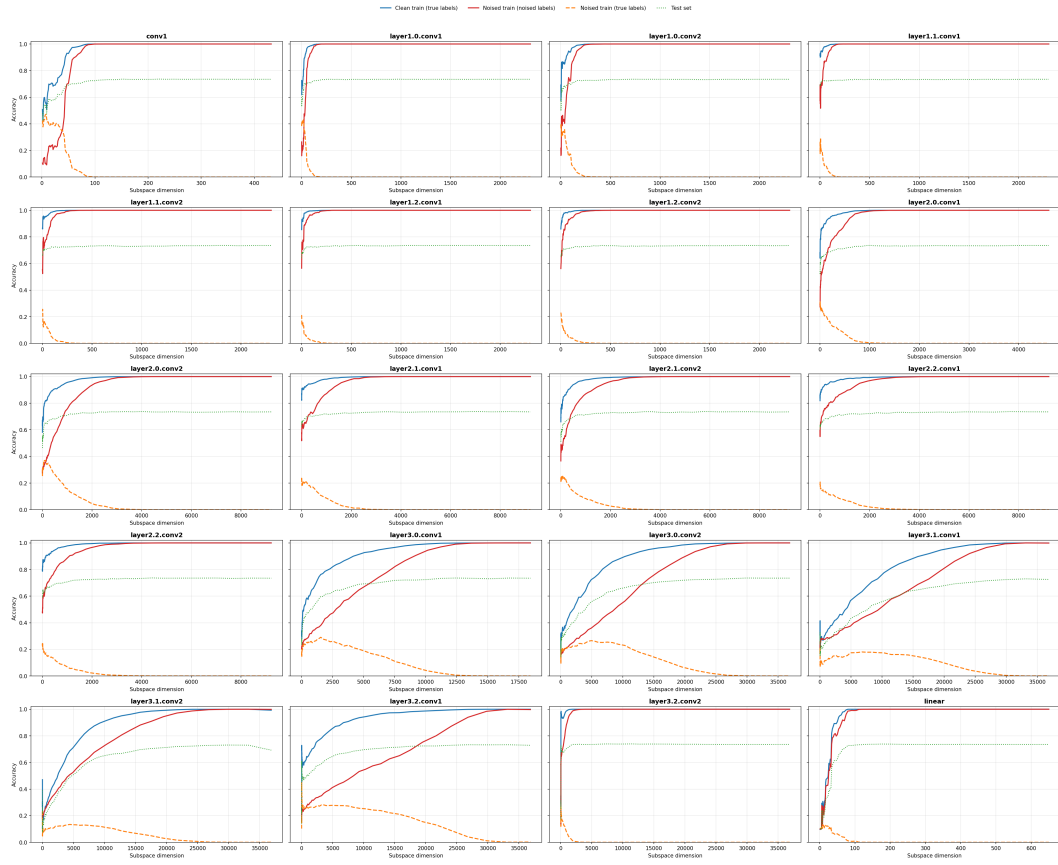


Figure 9: Fisher projections for every layer, in the noisy sample setting. Fisher computed on clean samples.

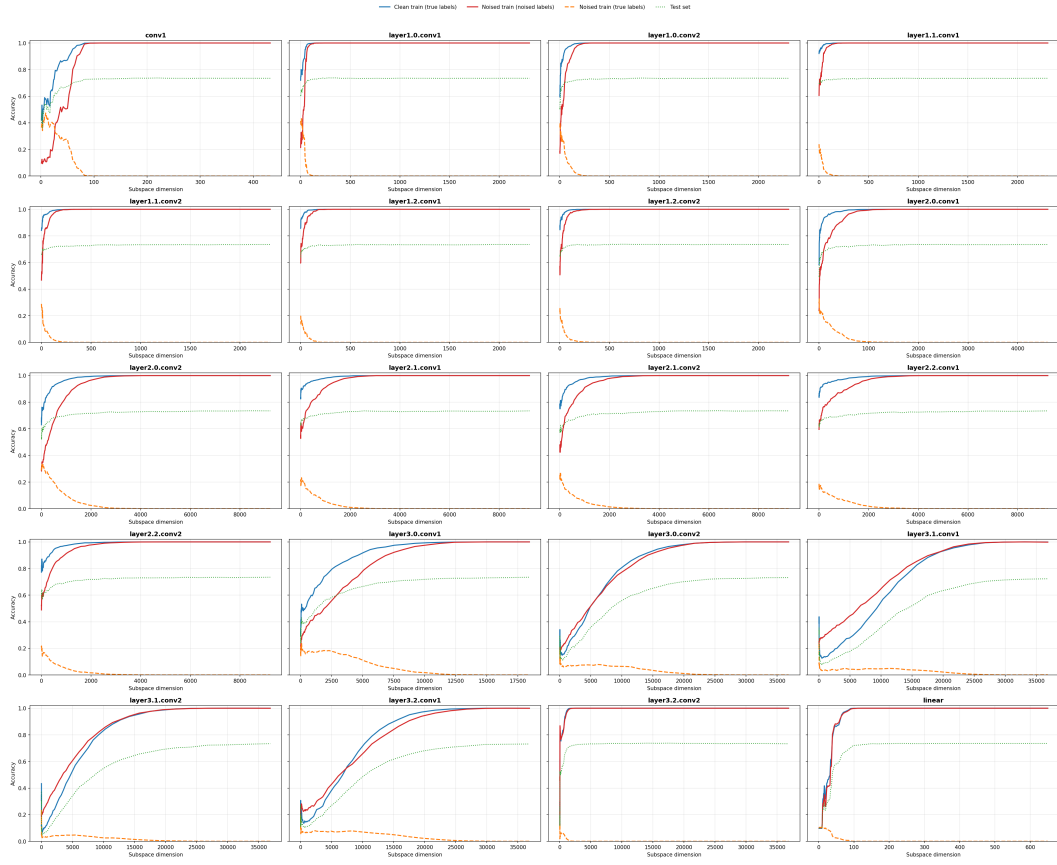


Figure 10: Fisher projections for every layer, in the noisy sample setting. Fisher computed on a subset of the full training set.

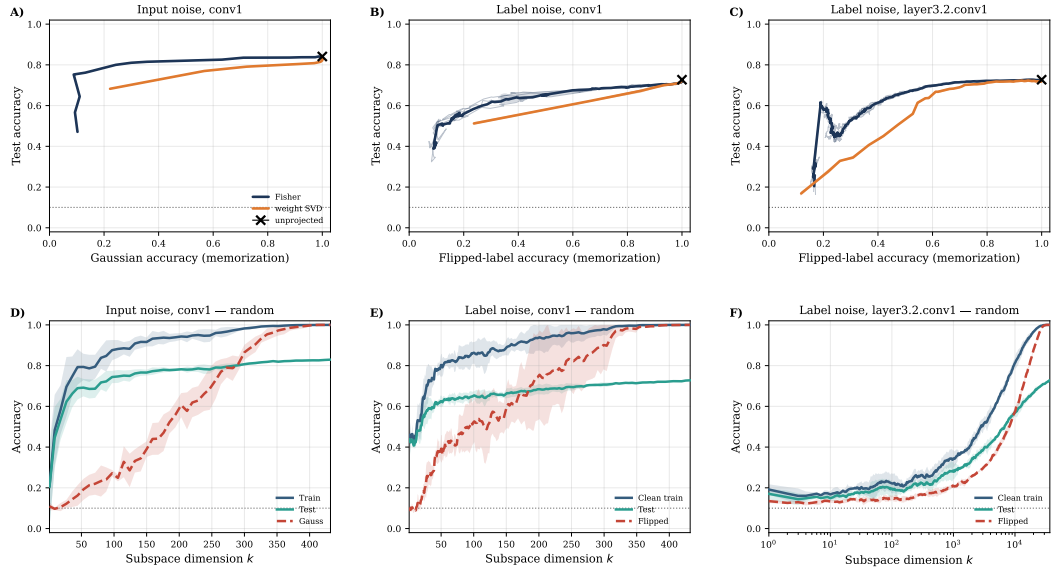


Figure 11: Control experiments for A)-C) SVD compression and D)-F) random basis projections.

C Barcode model derivations

Here, we derive the results stated in §5. We use the same notation as in the main text, $X = \begin{bmatrix} S \\ Z \end{bmatrix}$, $\tilde{Y} = D_{\tilde{y}}Y + \Xi$ and $\Pi_S^\perp = I_n - S^\top(SS^\top)^{-1}S$. We write $\Sigma_\xi = \text{diag}(a_1^2(1 - \tilde{\eta}_1^2), \dots, a_r^2(1 - \tilde{\eta}_r^2))$ for the noise covariance defined in Eq (5).

C.1 Estimating block coefficients

We obtain the estimators for the block coefficients by minimizing the squared loss

$$(\hat{\Theta}_s, \hat{\Theta}_z) = \min_{(\Theta_s, \Theta_z)} L(\Theta_s, \Theta_z) \quad \text{where} \quad L(\Theta_s, \Theta_z) = \|\tilde{Y} - \Theta_s S - \Theta_z Z\|_F^2. \quad (8)$$

At fixed Θ_z , the optimal signal block reads

$$\hat{\Theta}_s(\Theta_z) = (\tilde{Y} - \Theta_z Z)S^\top(SS^\top)^{-1}. \quad (9)$$

Substituting this expression in the loss yields

$$L(\hat{\Theta}_s(\Theta_z), \Theta_z) = \|\tilde{Y} - \Theta_z Z\|_{\Pi_S^\perp}^2. \quad (10)$$

Equating its derivative to zero gives

$$(\tilde{Y} - \hat{\Theta}_z Z)\Pi_S^\perp(Z\Pi_S^\perp)^\top = 0. \quad (11)$$

When $Z\Pi_S^\perp$ has full row rank d (i.e., $n-r \geq d$), this equation admits a unique solution

$$\hat{\Theta}_z = \tilde{Y}\Pi_S^\perp Z^\top(Z\Pi_S^\perp Z^\top)^{-1} = \Xi\Pi_S^\perp Z^\top(Z\Pi_S^\perp Z^\top)^{-1}, \quad (12)$$

The last equality results from the fact that $Y = D_a U = D_a D_\mu^{-1} S$ and $S\Pi_S^\perp = 0$. Hence, we have derived Eq (6).

C.2 Risk reduction from barcode deletion

We now derive the risk gain in Eq (7),

$$\Delta R = \mathbf{E} \|y - \hat{\Theta}_s s - \hat{\Theta}_z z\|^2 - \mathbf{E} \|y - \hat{\Theta}_s s\|^2 \quad (13)$$

$$= \mathbf{E} [-2(y - \hat{\Theta}_s s)^\top \hat{\Theta}_z z] + \mathbf{E} [\|\hat{\Theta}_z z\|^2] \quad (14)$$

$$= \mathbf{E} [\|\hat{\Theta}_z\|_F^2], \quad (15)$$

where the expectation is over the training set $(X, \tilde{Y})$ and a fresh test point (x, y) . In the above, we used the fact that the test barcode z is independent of all other random variables with $\mathbf{E}[z] = 0$ and $\mathbf{E}[zz^\top] = I_d$.

We evaluate this expression in two steps. First, we condition on the design (S, Z) and average over Ξ . Substituting Eq (12) in Eq (15) yields

$$\Delta R = \text{Tr} \mathbf{E} [\Xi \Pi_S^\perp Z^\top (Z \Pi_S^\perp Z^\top)^{-1} (Z \Pi_S^\perp Z^\top)^{-1} Z \Pi_S^\perp \Xi^\top] \quad (16)$$

$$= \text{Tr} \mathbf{E} [\mathbf{E}[\Xi^\top \Xi \mid S, Z] \Pi_S^\perp Z^\top (Z \Pi_S^\perp Z^\top)^{-1} (Z \Pi_S^\perp Z^\top)^{-1} Z \Pi_S^\perp] \quad (17)$$

$$= \text{Tr}(\Sigma_\xi) \text{Tr} \mathbf{E} [(Z \Pi_S^\perp Z^\top)^{-1}]. \quad (18)$$

In deriving the above, we use $\mathbf{E}[\Xi^\top \Xi \mid S, Z] = \text{Tr}(\Sigma_\xi) I_n$.

Second, we average over the Gaussian barcode Z . The projection $\Pi_S^\perp$ zeros out the r signal rows, leaving $m = n - r$ degrees of freedom. Since Z has iid zero-mean, unit-variance Gaussian entries, independent of S , the matrix $Z \Pi_S^\perp Z^\top$ follows a Wishart $_d(I_d, m)$ distribution.² For $m > d + 1$, the inverse-Wishart mean gives $\mathbf{E}[(Z \Pi_S^\perp Z^\top)^{-1}] = I_d/(m - d - 1)$ (see, e.g., Ref [45]). It immediately follows that

$$\Delta R = \text{Tr}(\Sigma_\xi) \frac{d}{n - r - d - 1}, \quad (19)$$

which concludes our derivation of Eq (7).

²To see this, let $U \in \mathbb{R}^{n \times m}$ have orthonormal columns spanning the range of $\Pi_S^\perp$, so $\Pi_S^\perp = UU^\top$. Then $Z \Pi_S^\perp Z^\top = (ZU)(ZU)^\top$ and $ZU \in \mathbb{R}^{d \times m}$ has iid standard Gaussian entries.

C.3 Partial-overlap prediction MSE

The risk gain in Eq (7) assumes clean separation between signal and barcode directions. The residual barcode covariance has upper sample-bulk edge $(1 + \sqrt{d/(n-r)})^2$, while a signal coordinate separates from that bulk only when its population variance satisfies $\mu_j^2 > 1 + \sqrt{d/(n-r)}$. Below this population threshold, empirical eigenvectors mix signal and barcode directions, and the projection rank K trades off signal loss against noise removal [42].

Let $XX^\top/n = \sum_k \hat{\lambda}_k u^{(k)} (u^{(k)})^\top$ with $(u^{(k)})^\top u^{(l)} = \delta_{kl}$ and $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \dots$, and $P_K = \sum_{k \leq K} u^{(k)} (u^{(k)})^\top$. The full estimator is $\hat{\Theta} = \Theta_{\bar{\eta}} + \Xi X^\top (XX^\top)^{-1}$, where $\Theta_{\bar{\eta}} = D_{\bar{\eta}} D_a D_\mu^{-1} (I_r \ 0_{r \times d})$ is the infinite-sample coefficient on corrupted targets. The post-hoc projected predictor is $\hat{y} = \hat{\Theta} P_K x$.

We define the cumulative alignment $A_{ii}(K) = \sum_{k \leq K} (u_i^{(k)})^2$ and barcode alignment $q_k = \sum_{l > r} (u_l^{(k)})^2$. Averaging over the fresh test point and the noise Ξ , conditional on X , gives the per-coordinate prediction MSE

$$\begin{aligned} R(K | X) &\equiv \frac{1}{r} \mathbf{E} \left[\|y - \hat{\Theta} P_K x\|^2 | X \right] \\ &= \frac{1}{r} \sum_{j=1}^r \mathbf{E} \left[\left(\frac{a_j}{\mu_j} (x - \bar{\eta}_j P_K x)_j - [\Xi X^\top (XX^\top)^{-1} P_K x]_j \right)^2 | X \right] \\ &= \frac{1}{r} \sum_{j=1}^r \frac{a_j^2}{\mu_j^2} \mathbf{E} \left[(x - \bar{\eta}_j [P_K x])_j^2 | X \right] + \frac{1}{r} \mathbf{E} \left[\|\Xi X^\top (XX^\top)^{-1} P_K x\|^2 | X \right] \\ &= \frac{1}{r} \sum_{j=1}^r \frac{a_j^2}{\mu_j^2} \left(\mu_j^2 - 2\bar{\eta}_j \mu_j^2 A_{jj}(K) + \bar{\eta}_j^2 \sum_{k, k' \leq K} u_j^{(k)} u_j^{(k')} (u^{(k)})^\top \Sigma_x u^{(k')} \right) \\ &\quad + \frac{\text{Tr}(\Sigma_\xi)}{rn} \sum_{k \leq K} \frac{(u^{(k)})^\top \Sigma_x u^{(k)}}{\hat{\lambda}_k} \end{aligned} \quad (20)$$

$$\begin{aligned} &= \frac{1}{r} \sum_{j=1}^r \frac{a_j^2}{\mu_j^2} \left(\mu_j^2 - 2\bar{\eta}_j \mu_j^2 A_{jj}(K) + \bar{\eta}_j^2 \left\{ \sum_{l=1}^r [P_K]_{jl}^2 \mu_l^2 + \sum_{l=r+1}^{r+d} [P_K]_{jl}^2 \right\} \right) \\ &\quad + \frac{\text{Tr}(\Sigma_\xi)}{rn} \sum_{k \leq K} \frac{(u^{(k)})^\top \Sigma_x u^{(k)}}{\hat{\lambda}_k} \end{aligned} \quad (21)$$

where $\Sigma_x = \text{diag}(\mu_1^2, \dots, \mu_r^2, 1, \dots, 1)$ is the population feature covariance. The cross-term vanishes since $\mathbf{E}[\Xi | X] = 0$. The variance term uses $P_K (XX^\top)^{-1} = n^{-1} \sum_{k \leq K} \hat{\lambda}_k^{-1} u^{(k)} (u^{(k)})^\top$.

The first term contains off-diagonal overlaps $(P_K)_{lj}$ with $l \neq j$, which are second-order small when Σ_x is diagonal and the empirical eigenvectors concentrate near coordinate axes. Setting these off-diagonal terms to zero and using the idempotence of P_K to replace $\sum_{l > r} (P_K)_{lj}^2$ by $A_{jj}(K)(1 - A_{jj}(K))$ gives

$$R_{\text{diag}}(K) = \frac{1}{r} \sum_{j=1}^r \left[a_j^2 (1 - \bar{\eta}_j A_{jj}(K))^2 + \frac{a_j^2 \bar{\eta}_j^2}{\mu_j^2} A_{jj}(K) (1 - A_{jj}(K)) \right] + \frac{\text{Tr}(\Sigma_\xi)}{rn} \sum_{k \leq K} \frac{(u^{(k)})^\top \Sigma_x u^{(k)}}{\hat{\lambda}_k}. \quad (22)$$

The first term in the square brackets is the squared mismatch between the clean target and the partially-projected shrunk predictor. The second is a mixing penalty from eigenvectors that partially align with signal coordinate j , amplified by $1/\mu_j^2$ for sub-threshold signals; it peaks at $A_{jj} = \frac{1}{2}$ and vanishes when each eigenvector is a pure coordinate axis (in which case $A_{jj} = 1$). The final term outside the square brackets is the noise variance retained by the projection.

An interior optimal rank requires heterogeneity or mixing. In the ideal separated limit, the retained eigenvectors are pure signal or pure barcode directions, so the mixing penalty vanishes and the optimal rule is to keep the signal directions and discard the barcode directions. Below the separation threshold, increasing K simultaneously recovers signal and admits noise; the optimum balances the two.

D Single-Sample Experiments

D.1 Model accuracy in single-sample projection experiments

Table 2: Bulk accuracy after sample projection. All accuracies are percentages. BatchNorm is reset after each projection and recalibrated by cumulative averaging on equal batches of 250 images. The $k = 0$ column is the matched unchanged-weight BatchNorm control.

Gaussian model						Noisy-label model (10% noise)					
	$k=0$	$k=1$	$k=2$	$k=4$	$k=9$		$k=0$	$k=1$	$k=2$	$k=4$	$k=9$
<i>Fisher source CIFAR-10</i>						<i>Fisher source uncorrupted</i>					
CIFAR-10 train (5k samples)	100.0	99.9	99.9	99.9	99.8	CIFAR-10 train (5k samples)	100.0	100.0	100.0	99.9	99.9
CIFAR-10 test	84.3	83.9	83.7	83.4	83.1	CIFAR-10 test	73.4	73.1	73.0	72.7	72.3
Gaussian set	100.0	100.0	100.0	100.0	99.9	Corrupted CIFAR (assigned labels)	100.0	100.0	100.0	99.9	99.8
						Corrupted CIFAR (true labels)	0.0	0.0	0.0	0.0	0.1
Same-class CIFAR (1k samples)	100.0	99.3	99.1	98.9	98.7	Same-class CIFAR (1k samples)	100.0	99.9	99.8	99.5	99.1
<i>Fisher source Gaussian</i>						<i>Fisher source corrupted</i>					
CIFAR-10 train (5k samples)	100.0	100.0	100.0	100.0	100.0	CIFAR-10 train (5k samples)	100.0	100.0	100.0	100.0	99.9
CIFAR-10 test	84.3	84.0	83.9	83.8	83.4	CIFAR-10 test	73.4	73.2	73.0	72.8	72.4
Gaussian set	100.0	98.4	96.1	90.5	75.6	Corrupted CIFAR (assigned labels)	100.0	100.0	100.0	99.9	99.8
						Corrupted CIFAR (true labels)	0.0	0.0	0.0	0.1	0.1
Assigned-label class (1k samples)	100.0	100.0	100.0	100.0	99.9	Assigned-class CIFAR (1k samples)	100.0	99.9	99.9	99.8	99.7
Source-class CIFAR (1k samples)	100.0	100.0	100.0	100.0	100.0	True-class CIFAR (1k samples)	100.0	100.0	100.0	100.0	99.8