Shared Memorization Structures in Transformers Revealed by Loss Curvature

Anonymous Author(s)
Affiliation
Address
email

Abstract

We characterize how memorization is represented in Transformer networks. We find that supervised memorization-removal models trained on a targeted set also suppress untargeted memorization, implying a shared representational structure for memorized data. Building on links between memorization and loss curvature, we show this structure is disentangled in weight space when expressed in the eigenbasis of the (K-FAC) Fisher information. Using this decomposition, we propose an **unsupervised** parameter-ablation method that outperforms a supervised method in suppression of memorization, yields more natural generations in LMs, and improves generalization in label-noisy ViTs. Our work expands the understanding of verbatim memorization in neural networks, and points to practical mitigation methods for suppressing it in trained models.

1 Introduction

Memorization of training data in neural networks has been extensively studied in the past but still eludes our full understanding. In generative models, memorization can lead to leakage of training data [Carlini et al., 2019, Shokri et al., 2017], which is possibly private or sensitive, or can violate copyright [Karamolegkou et al., 2023]. Although data filtering and deduplication are key in preventing memorization before training ends [Huang et al., 2024, Biderman et al., 2023], this is typically infeasible to do thoroughly [Goldblum et al., 2022], especially as today's systems scale the amount of data used to train them [Kaplan et al., 2020]. Mitigating memorization, therefore, remains a high priority for making models trustworthy and safe [Team et al., 2025].

From an interpretability perspective, understanding memorization is particularly interesting. How models generalize in some cases, but recite verbatim in others is both of practical and scientific interest to the community. Disentangling these behaviors can shed light on the internal representations that govern recall versus abstraction, helping us more accurately evaluate model capabilities. Removing or preventing memorization during training can make models safer and more reliable.

In this work, we study how memorization is structured in Transformer neural networks (ViTs and LMs). We observe that supervised methods for removing specific sets of memorized data generalize, and end up removing other memorized sequences not in the intended set, indicating that there is some *shared representational structure* controlling memorization. The most effective way we find to interpret this structure is from the lens of the curvature of the loss landscape w.r.t. weight matrices in the model, based on prior work connecting the loss curvature to generalization performance Foret et al. [2021], Hochreiter and Schmidhuber [1997], LeCun et al. [1989], Hassibi et al. [1993], Keskar et al. [2017] and memorization [Garg et al., 2024, Ravikumar et al., 2024, Jeon et al., 2024, Kim et al., 2023]. We find very distinct disentanglement in the eigenbasis of the Hessian of weight matrices between generalizing weight components, and those involved in memorization. Based on this understanding, we design an unsupervised model compression (Figure 1) technique for suppressing

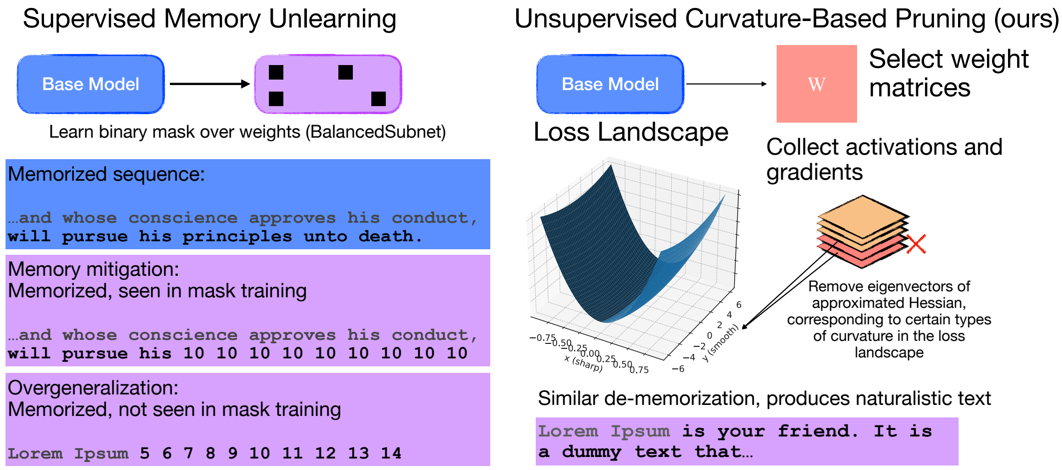

Figure 1: (Left) We find that existing supervised memory unlearning methods 'censor' memorized data that was not involved in the unlearning procedure, indicating shared structure involved in reciting memorized data. (Right) We interpret the basis for this structure in the weights of models, and use our understanding to design an unsupervised method for removing memorization based on directions in weight space that produce specific kinds of curvature in the loss landscape. Since our method does not involve gradient ascent, the edited model produces much more naturalistic generations. Prompts are in gray, generations in black.

memorized data that prunes directions in weight space that correspond to parameters likely involved in memorization. Despite requiring no labels of memorized data, our method is competitive with, and in some cases better, than a SotA supervised memorization removal technique (BalancedSubnet; Sakarvadia et al. [2025]). Since our method does not involve gradient ascent on memorized examples, models edited this way replace memorized examples with diverse but plausible generations, rather than random tokens (Figure 1), which may be preferable in some settings. Our contributions can be summarized as follows:

1. Whereas most previous work has studied memorization on a per-example basis, we demonstrate the existence of structure shared *across* memorized examples. In both ViTs and LMs, the eigenbasis of each layer's curvature (via K-FAC) isolates memorization directions from those supporting generalization.
2. We propose a new method for model compression that involves pruning this shared structure without supervision of what data is memorized. This method is outperforms a SotA supervised method in removal of verbatim memorization in diverse settings, and retains generation diversity in LMs.
3. We discuss the implications of shared memorization structure for other interpretability work, and provide practical considerations of dealing with memorization in neural networks based on our findings.

2 Methods

In this section, we operationalize how we measure memorization and the methods we use to understand it. This paper explores memorization in transformer models [Vaswani et al., 2017] trained for both image classification and language modeling. We analyze both modalities/tasks to provide a more robust account of memorization and to show the effectiveness of our method across diverse setups. Memorization takes a different form in both, however, and we outline how we measure them here.

2.1 Evaluating Memorization in Language Models

We use the OLMo2 family of models [OLMo et al., 2024] because they have openly accessible pretraining data and high performance on language modeling tasks. Previous work on evaluating memorization in LMs [Carlini et al., 2019, Huang et al., 2024, Shokri et al., 2017, Carlini et al., 2022] sample sequences from the pretraining data using some prefix P, and determine that a sequence is memorized if greedy decoding produces some suffix S. We use $|P| = 64$ and $|S| = 48$ for the experiments in this paper, unless noted otherwise. Our metrics for measuring memorization are **strict accuracy** which measures the proportion of memorized sequences for which the model generates S exactly given P, **loose accuracy**, which measures the fraction of examples with token-level Levenshtein similarity $\geq$0.75 to the target (includes near-copies/paraphrases), and **avg. Levenshtein** which measures the mean of d/$|S|$ per example, where d is token edit distance and $|S|$ is suffix length. 0 = identical, 1 = maximally different.

2.1.1 BalancedSubnet

Sakarvadia et al. [2025] perform a comprehensive analysis on suppression of memorization in LMs across various different techniques. They introduce a simple and high-performing method for suppressing memorization called BalancedSubnet (BSN). The approach in BSN is to learn a binary mask over individual parameters in MLP matrices of models while *increasing* loss on a set of memorized sequences (the *forget* set), while *decreasing* loss on a set of non-memorized sequences (the *retain* set). The intuition is to learn the minimal binary mask that maximizes loss on the forget set without destroying general model capabilities. We implement this method for OLMo to better understand memorization in LMs and to compare to our proposed approach. We train a binary mask for 2 epochs on 500 sequences with prefix length 64 and suffix length 48. Hyperparameters can be found in Appendix C.

2.2 Evaluating Memorization in Vision Models

In image classification models, memorization has been well studied, and there are simple recipes for producing models that memorize specific images. We train a family of 86M parameter ViT-Base models [Dosovitskiy et al., 2020] with 16x16 image patches at image resolution 224x224. We follow Dosovitskiy et al. [2020] training recipe on the ILSVRC 2012 ImageNet dataset [Russakovsky et al., 2015]. In order to control memorization, we train ViT variants where a subset of training images have randomly assigned ('noised') labels. The only way for a model to reduce the loss on these images is to memorize these input-label pairs exactly. This is a standard setup for evaluating memorization in image classifiers [Zhang et al., 2017]. Our default for evaluation is to train with 10% noised labels for 300 epochs. This model achieves a top-1 accuracy on the validation set of 68.7%. When training with no noise, our model achieves 77.2% top-1 accuracy.

2.3 Background on Loss Curvature and K-FAC

Following Martens and Grosse [2015], Foret et al. [2021], we study memorization and generalization through the lens of loss curvature. Like previous investigators, we consider the loss as a function of the model's weights, and hypothesize that sharp curvature indicates directions in weight space used for memorization, while flatter directions are used for generalization. Mathematically, the curvature of the loss landscape is captured by the Hessian $\mathbf{H} = \nabla_\theta^2 L(\theta)$, where L is the loss function and θ is the vector of flattened model weights. Each eigenvalue of $\mathbf{H}$ gives the amount of curvature along its corresponding eigenvector. Practically, though, $\mathbf{H}$ is not tractably computable for any but the smallest models, as its size is quadratic in the number of model weights. In order to approximate the structure of $\mathbf{H}$, we turn to the Kronecker-Factored Approximate Curvature (K-FAC) introduced in Martens and Grosse [2015]. K-FAC was introduced as an efficient natural-gradient method that approximates the Fisher Information Matrix (FIM), providing a structured approximation to the loss curvature without forming the full Hessian. For a model trained with softmax cross-entropy loss, the relationship of the FIM $\mathbf{F}$to the curvature of parameters is given by:

$$\mathbf{F} = \mathbb{E}_D[\nabla_\theta \log p_\theta(y \mid x, \theta) \nabla_\theta \log p_\theta(y \mid x, \theta)^T] = \mathbb{E}_D[\nabla_\theta^2(-\log p_\theta(y|x))]$$

Here, D is a dataset consisting of input-label pairs (x, y), and $p_\theta(y \mid x)$ is the model's predicted label distribution for input x. For an individual matrix $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ with incoming activations a and backpropagated gradients g, K-FAC gives an easily computable approximation to a block of $\mathbf{F}$:

$$\mathbf{F} \approx \mathbf{G} \otimes \mathbf{A} = \mathbb{E}[gg^T] \otimes \mathbb{E}[aa^T] \tag{1}$$

In words, this is the Kronecker product of the (uncentered) second-moment matrices of the activations going into the layer and the gradients coming out. Thus, K-FAC factorizes the Fisher block as the

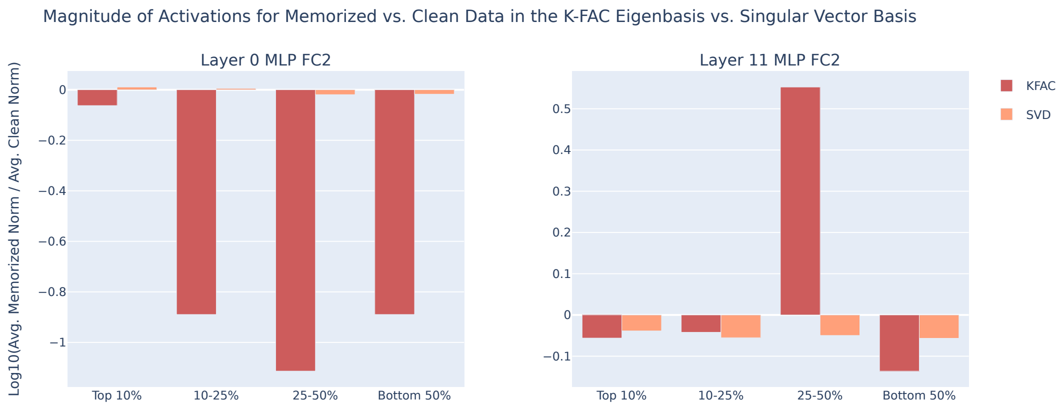

Figure 2: Comparison of K-FAC vs. SVD sensitivity to memorized vs. clean data for the first and last MLP down-projections in ViT-Base trained with 10% label noise. As measured by the magnitude of activation of memorized vs. clean data, the K-FAC eigenbasis identifies **directions in weight space that fire 10x more strongly on clean data (left) and 3.5x more strongly for memorized data (right)**. For comparison, we include projections for projections on singular vectors of the weight matrices, where the balance is about even (i.e., no disentanglement).

Average Levenshtein Distance ↑		nDCG@10 ↑
Mem-Train	Mem-Heldout	Non-Memorized
0.93	0.86	0.9134

Table 1: Avg. Levenshtein distance on memorized and nDCG@10 on non-memorized heldout sequences after BalancedSubnet show that the edit generalizes to memorized sequences that were not seen in training, while affecting non-memorized sequences much less. We hypothesize this is due to shared structure in weights that is used in recitation of memorized data more generally.

product between $\mathbf{A} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$ and $\mathbf{G} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{out}}}$. We can compute the eigendecomposition of our approximated average Hessian using the eigendecomposition of $\mathbf{A}$ and $\mathbf{G}$ (see Appendix A). Rather than taking y to be supervised labels, we sample $\hat{y}$ from the model's predicted label distribution. Not only is this important for computing the correct FIM[Martens and Grosse, 2015], but it also makes our method entirely unsupervised.

3 Memorized Representations Share Common Structure

This section presents evidence for our claim that memorized data learned by neural networks shares a common representational signature, and that we can find this signature in the weights of models. We first observe that BalancedSubnet [Sakarvadia et al., 2025], a supervised method for suppressing a predetermined set of verbatim memorized sequences from a language model generalizes to nontrivial memorized sequences that were not seen in training of the BSN subnetwork. Table 1 shows that BSN drops accuracy on 115 heldout memorized sequences that are different from the data used to train the subnetwork mask; an example is shown in 1. Since BSN does not harm more general language-modeling faculties, this generalization is unlikely to arise from generally destructive lesioning. Instead, we hypothesize that diverse memories, both in and out of the 'forget' set share a common substrate, so that forgetting the targeted memories disrupts other memories too. In the following section, we interpret this behavior and explore the hypothesis that shared weight components control recitation of memorized sequences.

3.1 Disentangling Memorization Parameter Components in the K-FAC Eigenbasis

Building on prior work like [Hochreiter and Schmidhuber, 1997, Foret et al., 2021, Kim et al., 2023], we hypothesize that the structure shared across diverse memories takes the form of directions in weight space, and that we can use the K-FAC machinery introduced in 2.3 to locate these directions.

Setup Equation 1 gives us a proxy for the curvature of the loss around a given weight matrix $\mathbf{W}$ computed as the Kronecker product of $\mathbf{A}$, the covariances of $\mathbf{W}$'s incoming activations, and $\mathbf{G}$, the covariances of $\mathbf{W}$ output-side gradients. Materializing this product is computationally infeasible, but hypothetically, the eigendecomposition of it will give us a basis ordered by how much the loss landscape curves in each direction. We can project hidden states in the model (right before $\mathbf{W}$) onto subsets of this basis and measure the norm, in order to understand . If $\mathbf{A}$ has eigenpairs $(\lambda, \mathbf{v})$ and G has eigenpairs $(\mu, \mathbf{u})$ (i.e., the ith eigenvalue and eigenvector), then an eigenvalue of $\mathbf{F}$ is $\kappa = \lambda\mu$ and the corresponding eigenvector is $\mathbf{z} = \mathbf{v} \otimes \mathbf{u}$. Note that for a weight matrix $\mathbf{W} \in \mathbb{R}^{m \times n}$, then $\mathbf{z} \in \mathbb{R}^{mn}$. Let $\mathbf{Z} := \text{unvec}(\mathbf{z}) \in \mathbb{R}^{m \times n}$. For a hidden state $\mathbf{x}$, we take the L2 norm of $\mathbf{Zx}$ to measure this mode-specific response/sensitivity of the output to moving $\mathbf{W}$ along the eigenvector $\mathbf{z}$. We hypothesize that memorized and non-memorized data have different activations along different percentile bands of the curvature spectrum, indicating disentanglement in this basis. We sort the eigenvalues of the approximate Fisher by $\lambda_i \mu_j$ and sample eigenvectors[1] from the top 10%, 10-25%, 25-50%, and bottom 50% of eigenvectors, computing $||\mathbf{Zx}||_2$ for each sampled $\mathbf{Z}$ over 10k hidden states. We report the average norm for data in each of these bands. For computational efficiency, we compute these values with the ViT-Base model we train with 10% label noise (rather than the billion param. LMs we use).

Results We hypothesize that different parts of the curvature spectrum will have different activations for generalizing vs. memorizing data. We study the ViT model with implanted memories described in 2.2, and report MLP layers 0 and 11 (down-projection matrices), which we find to have the sharpest distinctions in activation, as measured by the ratio of activation magnitude of memorized images over non-memorized images. Figure 2 shows that non-memorized data activates up to 10x more strongly to the lower eigenvalue band in MLP 0. In MLP 11, activations for the 25-50th percentile bands activate about 3.5x more strongly for memorized images. As a baseline, we also computed the SVD of these MLP weight matrices, We find that this separation does not occur in the spectrum of the SVD of these weight matrices, indicating that viewing the weights in the basis of directions with more/less curvature uniquely separate these values. Prior work demonstrates that for a single example, the curvature of the loss is higher for memorized examples [Garg et al., 2024, Ravikumar et al., 2024, Jeon et al., 2024]. Our finding is consistent with that finding, but differs because we are looking at a proxy for the *average* curvature over a dataset. A direction in weights that is high curvature for a single memorized example is low curvature for most other examples, whereas directions used for many examples (non-memorized) have some amount of curvature across the whole dataset.

4 Suppressing Memorization with K-FAC Weight Projections

If there are disentangled representations for memorized and non-memorized data within the parameters of models, then we should be able to remove them without negatively impacting generalization performance. In this section, we propose a model compression strategy that involves ablating directions in weight space that are involved in the recitation of memorized data. In Section 3.1, we show that the eigenbasis of the Fisher block at some layers can strongly disentangle memorized representations (see also Appendix B). These computations do not involve a dataset labeled for memorized sequences, but do require a sample of (unlabeled) data to collect activations and gradients for a layer. Following results in Figure 2, we propose a pruning method based on truncating the bottom k% of eigendirections of a layer's Fisher block. Intuitively, the top eigendirections of the Fisher block are directions in weight space that, when perturbed, affect the loss substantially across a dataset, and the bottom eigendirections affect the loss little, or only matter for a few examples. Pruning the bottom-most directions may seem inconsistent with prior work Garg et al. [2024], LeCun et al. [1989], Hassibi et al. [1993], which shows that memorized data has *higher* curvature than non-memorized data at an instance level. The key intuition is that since K-FAC gives us a proxy for average curvature across the dataset, highly curved directions for memorized data are actually flat for most inputs, and the directions that change loss consistently across examples correspond to weights for generalizing features (firing for the majority of data).

To ablate lower eigenvalue directions, we sample from the top eigenvalues of the Hessian (computed as the products between eigenvalues of $\mathbf{A}$ and $\mathbf{G}$), and retain the corresponding eigenvector directions, continuing until the total energy of eigenvalues surpasses the a threshold percentage of the total

[1]We sample instead of computing each one, since there are millions.

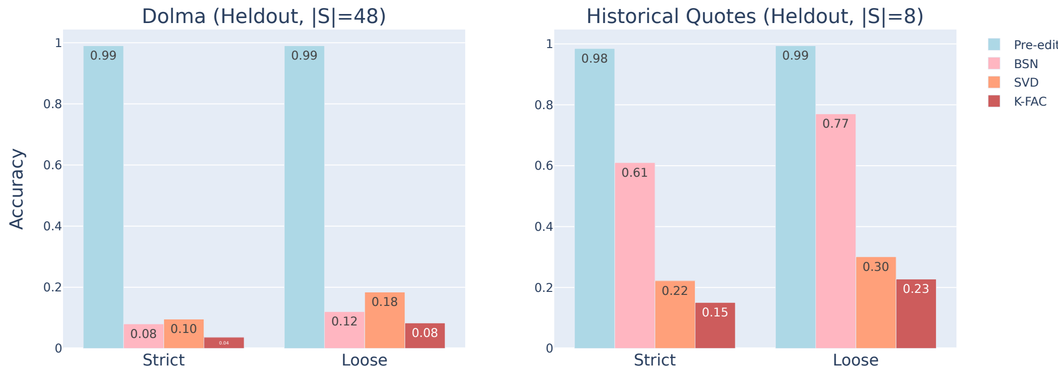

Figure 3: Editing results for BalancedSubnet, and truncating weight matrices according to SVD or K-FAC directions in OLMo2 7B. K-FAC has the best memorization suppression, going from 99% to 4% strict matching, and extends the best to a difficult quotes domain.

	Memorized Data Variation Average Levenshtein Distance		Non-memorized Data Preservation nDCG@10
	Dolma	Quotes	
Pre-edit	0.01	0.01	1.00
BSN	0.86	0.21	0.91
SVD	0.64	0.58	0.90
K-FAC	0.72	0.64	0.91

Table 2: Additional statistics for editing metrics showing variation from memorized suffixes (measured by Levenshtein distance) and non-memorized data's preservation of the unedited model's top 10 next token predictions. All metrics retain faithfulness to non-memorized data. BSN transfers the least to the quotes dataset, and K-FAC is the most balanced. For all metrics, higher is better.

energy.In general, we keep the top k=90% of energy of the eigenspectrum. We then project out the remaining directions as follows:

$$\begin{aligned}
\mathbf{P_A} &= \mathbf{U_A}[:, :\text{keep_idxs_A}] \\
\mathbf{P_G} &= \mathbf{U_G}[:, :\text{keep_idxs_G}] \\
\hat{\mathbf{W}} &= \mathbf{P_G}\mathbf{P_G^T}\mathbf{W}\mathbf{P_A}\mathbf{P_A^T}
\end{aligned}$$

4.1 Experimental Setup

We test our compression procedure on both OLMo-2 LMs and ViT-base on an image classification task. We search layer-wise for the best single-layer editing strategy, and additionally perform a coarse-grained search for the best multi-layer edit. We expect that since our procedure edits out low variance directions in weight space, we may be able to disrupt them by truncating low variance singular values as well, which would be a cheaper and data-free alternative. So, in addition to the K-FAC projection edit we described in the previous section, we also run a baseline testing whether keeping the top [1, 2, 5, 10, 20, 30, 50] percent of singular values from the same layers can perform similarly to K-FAC. For image models, we use a subset of 10k images from the ImageNet training set, and for OLMo LMs, we use 2M tokens from the Dolma training corpus. In fully unsupervised settings, our results suggest editing the earliest few layers, and in some cases include the last layer for best expected results.

4.2 Results

4.2.1 Language Models

We evaluate our editing procedures on 125 memorized sequences from Dolma (not used to calculate $\mathbf{A}$ and $\mathbf{G}$) with a suffix length ($|S|$) of 48 and a dataset of 125 memorized historical quotes with a suffix

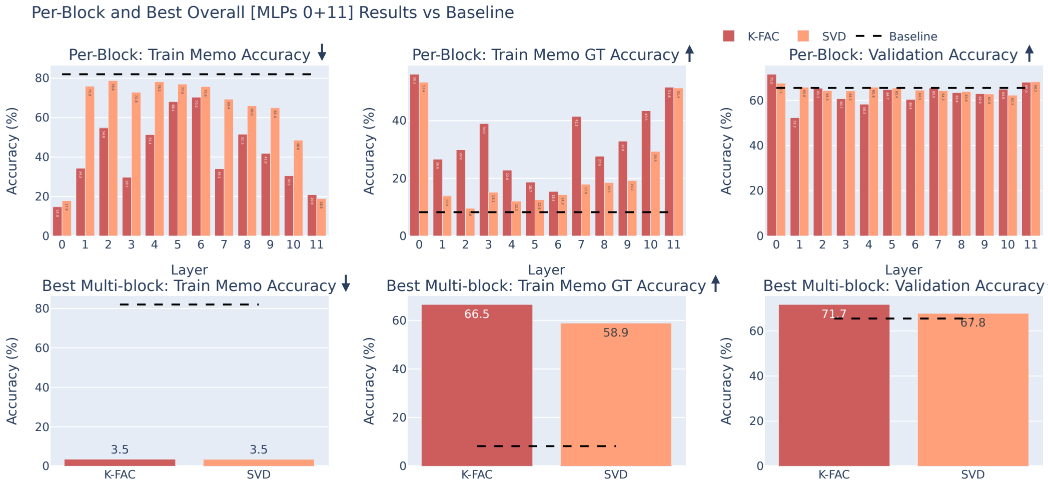

Figure 4: Comparison of K-FAC compression and SVD per MLP block (top) and with the best configuration (bottom) in a ViT model. We find that K-FAC compression generally outperforms SVD, and the best results (compressing layers 0 and 11 simultaneously) aligns with the results in §3.1, where these layers showed the greatest disentanglement between memorized data and generalizing data. Note that with K-FAC we are able to effectively remove memorization while *substantially improving* generalization performance (validation), and recovering more of the ground truth label on the previously memorized set than SVD.

length of 8. For BalancedSubnet, we train on 500 memorized sequences from Dolma and evaluate on the 125 heldout memorized Dolma sequences and 125 quotes. The quotes are included because they are more challenging due to probable high repetition in the pretraining data, and evaluating on the next 8 tokens. The quotes are also truly heldout: In all cases, the quotes were not used to adjust **any** hyperparameters in K-FAC, SVD, or BSN. This setting also tests both length generalization and generalization to a domain that isn't web text. For K-FAC pruning, we keep the top 80% of energy in the first three MLPs; for SVD, we use the top 20% singular values for these layers. Figure 3 shows our results on all procedures. We find that K-FAC outperforms BalancedSubnet on recitation of Dolma sequences, producing half as many memories, and outperforms all other methods on historical quotes, indicating that K-FAC pruning find the most general weights responsible for memorization, despite not training with any signal on what was memorized. Table 2 shows token level variation through Levenshtein distance on memorized heldout sequences. Unsuprisingly, BSN poduces the most distinct text (since it generates random numbers when it detects in-domain memorization; Figure 1), but fails to generalize out of its training distribution: on quotes, BSN gets only 0.21 average Levenshtein, while SVD gets 0.58, and K-FAC does the best with 0.64. On heldout, non-memorized sequences, the top-10 next token predictions are very close to the unedited model's set, as measured by nDCC@10. The generations resulting after K-FAC and SVD editing on these matrices are much more naturalistic than BSN, since we do not involve gradient ascent. We include sample generations from memorized and non-memorized prompts in Appendix D.

4.2.2 Vision Transformers

Figure 4 shows the results for editing ViT-Base with 10% training noise in various settings. On a per-layer basis, we see that pruning the earliest and latest layers provides the best results across the board. For both K-FACWe achieve the best performance when we prune MLPs 0 and 11 simultaneously, driving memorization performance down to 3.5% from over 80%. K-FAC also *increases* the validation accuracy over 4% from 67% to 71.7%, while SVD only increases performance around 1%. If we have successfully targeted memorized features, then we should see that the images that were memorized should switch to predicting their ground truth (GT) labels. K-FAC successfully raises the ground truth accuracy up to 66.5% while SVD reaches 58.9%.

4.3 Final Remarks

Our results provide a deeper understanding of memorization in neural networks trained in two distinct settings, and show that recitation of memorized data shares common structure in the weight space of models, especially in the early layers (consistent with [Stoehr et al., 2024, Maini et al., 2023]). The method we propose could still be better understood and improved upon. We discuss practical recommendations and important limitations of our results in the following sections.

5 Related Work

Memorization in neural networks has been explored from many angles, and often as an overfitting/generalization problem, making it one of the most widely studied in deep learning. With increasing LLM scale [Kaplan et al., 2020], there is building interest in understanding the sheer amount of data such models memorize [Carlini et al., 2022, Morris et al., 2025, Karamolegkou et al., 2023]. Relatedly, there is great interest in compressing models using low-rank structure in gradients [Zhao et al., 2024] in order to collapse weight matrices into low-rank approximations [JAISWAL et al., 2025]; also using the SVD [Sharma et al., 2023]. Memorization and generalization in terms of spectral dynamics has also been explored [Yunis et al., 2024]. We do not explore this direction deeply, but our K-FAC-based method can be used to reduce the rank of weights (and SVD always reduces the rank). Other work on memorization focuses on the question of whether memories can be localized in model weights [Maini et al., 2023, Chang et al., 2024]. Our work suggests, aligning to previous work [Hase et al., 2023], that memorization is hard to pinpoint (and likely highly distributed), but distinctly loss-curved directions related to recitation of memorized data can be localized to some (early/late layers). This connects to interpretability work on finding distinct processes involved in fact retrieval within models [Geva et al., 2021, Gur-Arieh et al., 2025, Meng et al., 2022, Dai et al., 2022, Rajamanoharan et al., 2023, Merullo et al., 2024, Menta et al., 2025]. Further work is needed to combine views differentiating memorization from structured mechanisms for fact retrieval.

6 Discussion

We have taken a step towards understanding memorization of training data in both ViT and LM Transformer models,and find consistent structure in the weight space, especially in early layers, that seems to be shared across memories. This is surprising, perhaps, as knowledge in neural networks is often considered highly distributed, an idea supported by classic studies on “graceful degradation" to neuron ablations [Rumelhart et al., 1986], and more recently by interpretability work on “microfeatures" [Rajamanoharan et al., 2023]. While our results do not make strong claims about ‘localization’ of memories to any particular point, we find that the curvature-basis of weights nicely disentangles parameter directions involved in memorization,indicating significant spatial organization of memory storage.

6.1 Practical Considerations for De-Memorizing Models

Our work is motivated in understanding the representational differences between memorizing and generalizing components of models, and our pruning procedure is designed to demonstrate the precision of our findings. While our results show that this approach is effective across models, we do not claim that this method will fully remove memorized data from a model. Unlearning methods are known to ‘suppress’ rather than fully remove the target domain [Hong et al., 2024, Lee et al., 2025, Barez et al., 2025]. As our method relates to unlearning, K-FAC pruning suffers from the same issues. Lee et al. [2025] discuss making unlearning more robust by distilling the model after applying unlearning to it, which could work in our setting.

When considering editing a model, it can vary what the desired properties are. In some cases, the more targeted appraoch of BalancedSubnet [Sakarvadia et al., 2025] that censors generations predicted as memorized may be preferable. Our approach aims at the most general purpose treatment of memorization in models; suppressing it in the widest range of settings. However, we also aim to maintain naturalistic generations after the edit. For some uses, this is desirable. In others, subtly wrong but believable outputs are troublesome. Therefore, we do not suggest that any one method presented in this paper is always the most effective in every setting.

6.2 Memorization and Policy around Model Disgorgement

Memorization presents a real problem for deploying neural networks into the world that we can trust. As models become more widely used across and ubiquitous in daily life, their responsible use and enforcement of user protections becomes more important. In order to prevent leaks of harmful training data, governing bodies may require re-training or model deletion, but our current understanding of information representation within models does not currently afford knowing when information might fully be removed from a model, or if deletion is necessary. Regulators have already shown they will reach past raw datasets to the models themselves. In the United States, the Federal Trade Commission has repeatedly ordered "algorithmic disgorgement," requiring companies to delete models trained on unlawfully obtained or processed data [Goland, 2022].

The temporary withdrawal and later "safety-revised" re-release of LAION-5B [Schuhmann et al., 2022] after the discovery of links to illegal content illustrates how quickly a widely used corpus can become legally non-deployable. When a foundation dataset is implicated, models trained with some subset of it are implicated as well. This creates cascading costs: product freezes, re-training, and potential disgorgement.

It is clear that current post-training/mitigation methods are not enough to prevent verbatim production of training data [Nasr et al., 2025]. Formal machine unlearning remains immature at the scale and heterogeneity of modern pretraining. We see interpretability as one way to address complications related to data leakage in a way that does not require the exorbitantly expensive and environmentally harmful costs of retraining an affected model. our work presents a step toward better understanding of the representation of memorized data within the weights of a network. Advancing this understanding will allow us to more accurately judge the permanence of the imprint training data leaves on a model, and what interventions may be necessary to remove it.

7 Conclusion

We have shown curvature of the loss around weight matrices in Transformer models identifies structure across memorized examples in LMs and ViTs. This work explores using this insight to design a simple, label-free way to prune memorization across different domains, and shows this method is effective at reducing verbatim memorization while maintaining naturalistic generations (in LMs) and good downstream performance (in ViTs). Future work could explore improvements to our proposed pruning method, extend the method to more aggressively shrink model parameter counts, or better understand generalization in models.

References

Fazl Barez, Tingchen Fu, Ameya Prabhu, Stephen Casper, Amartya Sanyal, Adel Bibi, Aidan O'Gara, Robert Kirk, Ben Bucknall, Tim Fist, et al. Open problems in machine unlearning for ai safety. *arXiv preprint arXiv:2501.04952*, 2025.

Stella Biderman, Usvsn Prashanth, Lintang Sutawika, Hailey Schoelkopf, Quentin Anthony, Shivanshu Purohit, and Edward Raff. Emergent and predictable memorization in large language models. *Advances in Neural Information Processing Systems*, 36:28072–28090, 2023.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX security symposium (USENIX security 19)*, pages 267–284, 2019.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations*, 2022.

Ting-Yun Chang, Jesse Thomason, and Robin Jia. Do localization methods actually localize memorized data in llms? a tale of two benchmarks. In *NAACL-HLT*, 2024.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, 2022.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, G Heigold, S Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021.

Isha Garg, Deepak Ravikumar, and Kaushik Roy. Memorization through the lens of curvature of loss function around samples. In *International Conference on Machine Learning*, pages 15083–15101. PMLR, 2024.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2021.

Joshua A Goland. Algorithmic disgorgement: Destruction of artificial intelligence models as the ftc's newest enforcement tool for bad data. *Rich. JL & Tech.*, 29:1, 2022.

Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Mądry, Bo Li, and Tom Goldstein. Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1563–1580, 2022.

Yoav Gur-Arieh, Clara Suslik, Yihuai Hong, Fazl Barez, and Mor Geva. Precise in-parameter concept erasure in large language models. *arXiv preprint arXiv:2505.22586*, 2025.

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. *Advances in Neural Information Processing Systems*, 36:17643–17668, 2023.

Babak Hassibi, David G Stork, and Gregory J Wolff. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, pages 293–299. IEEE, 1993.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 9(1):1–42, 1997.

Yihuai Hong, Lei Yu, Haiqin Yang, Shauli Ravfogel, and Mor Geva. Intrinsic evaluation of unlearning using parametric knowledge traces. *arXiv preprint arXiv:2406.11614*, 2024.

Jing Huang, Diyi Yang, and Christopher Potts. Demystifying verbatim memorization in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10711–10732, 2024.

AJAY KUMAR JAISWAL, Yifan Wang, Lu Yin, Shiwei Liu, Runjin Chen, Jiawei Zhao, Ananth Grama, Yuandong Tian, and Zhangyang Wang. From low rank gradient subspace stabilization to low-rank weights: Observations, theories, and applications. In *Forty-second International Conference on Machine Learning*, 2025.

Dongjae Jeon, Dueun Kim, and Albert No. Understanding memorization in generative models via sharpness in probability landscapes. *CoRR*, 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Antonia Karamolegkou, Jiaang Li, Li Zhou, and Anders Søgaard. Copyright violations and large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7403–7412, 2023.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017.

Young In Kim, Pratiksha Agrawal, Johannes O Royset, and Rajiv Khanna. On memorization and privacy risks of sharpness aware minimization. *CoRR*, 2023.

Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

Bruce W. Lee, Addie Foote, Alex Infanger, Leni Shor, Harish Kamath, Jacob Goldman-Wetzler, Bryce Woodworth, Alex Cloud, and Alexander Matt Turner. Distillation robustifies unlearning, 2025. URL `https://arxiv.org/abs/2506.06278`.

Pratyush Maini, Michael C Mozer, Hanie Sedghi, Zachary C Lipton, J Zico Kolter, and Chiyuan Zhang. Can neural network memorization be localized? In *Proceedings of the 40th International Conference on Machine Learning*, pages 23536–23557, 2023.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417. PMLR, 2015.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022.

Tarun Ram Menta, Susmit Agrawal, and Chirag Agarwal. Analyzing memorization in large language models through the lens of model attribution. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 10661–10689, 2025.

Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. Language models implement simple word2vec-style vector arithmetic. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5030–5047, 2024.

John X Morris, Chawin Sitawarin, Chuan Guo, Narine Kokhlikyan, G Edward Suh, Alexander M Rush, Kamalika Chaudhuri, and Saeed Mahloujifar. How much do language models memorize? *arXiv preprint arXiv:2505.24832*, 2025.

Milad Nasr, Javier Rando, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A. Feder Cooper, Daphne Ippolito, Christopher A. Choquette-Choo, Florian Tramèr, and Katherine Lee. Scalable extraction of training data from aligned, production language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=vjel3nWP2a`.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024.

Senthooran Rajamanoharan, Neel Nanda, János Kramár, and Rohin Shah. Fact finding: How to think about interpreting memorisation (post 4) — AI alignment forum, 2023. URL `https://www.alignmentforum.org/posts/JRcNNGJQ3xNfsxPj4/fact-finding-how-to-think-about-interpreting-memorisation`.

Deepak Ravikumar, Efstathia Soufleri, Abolfazl Hashemi, and Kaushik Roy. Unveiling privacy, memorization, and input curvature links. In *International Conference on Machine Learning*, pages 42192–42212. PMLR, 2024.

David E Rumelhart, James L McClelland, PDP Research Group, et al. *Parallel distributed processing, volume 1: Explorations in the microstructure of cognition: Foundations*. The MIT press, 1986.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Mansi Sakarvadia, Aswathy Ajith, Arham Mushtaq Khan, Nathaniel C Hudson, Caleb Geniesse, Kyle Chard, Yaoqing Yang, Ian Foster, and Michael W Mahoney. Mitigating memorization in language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in neural information processing systems*, 35:25278–25294, 2022.

Pratyusha Sharma, Jordan T Ash, and Dipendra Misra. The truth is in there: Improving reasoning in language models with layer-selective rank reduction. In *The Twelfth International Conference on Learning Representations*, 2023.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

Niklas Stoehr, Mitchell Gordon, Chiyuan Zhang, and Owen Lewis. Localizing paragraph memorization in language models. *arXiv preprint arXiv:2403.19851*, 2024.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

David Yunis, Kumar Kshitij Patel, Samuel Wheeler, Pedro Savarese, Gal Vardi, Karen Livescu, Michael Maire, and Matthew R Walter. Approaching deep learning through the spectral dynamics of weights. *arXiv preprint arXiv:2408.11804*, 2024.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient llm training by gradient low-rank projection. In *Forty-first International Conference on Machine Learning*, 2024.

A Primer on the Eigendecomposition of A and G

This section provides background on how to think about the eigenvectors and eigenvalues of the Hessian, as approximated by the K-FAC factorization $\mathbf{F} \approx \mathbf{G} \otimes \mathbf{A}$. For a given weight matrix, recall that $\mathbf{A}$ is the covariance matrix of the activations going into it, and that $\mathbf{A} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$. $\mathbf{G}$ is the covariance matrix of the gradients on the output side of the matrix, and $\mathbf{G} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{out}}}$.

Notice that we have $d_{\text{in}} * d_{\text{out}}$ eigenpairs in the Hessian. The approximate eigenvalues of the FIM are the products between each of the eigenvalues of the $\mathbf{G}$ and $\mathbf{A}$ matrices from K-FAC, and the corresponding eigenvectors are the Kronecker products between the eigenvectors of $\mathbf{G}$ and $\mathbf{A}$.

B Further Eigenspectrum Analysis for ViT Models

In this section we will describe how we computed the eigenspectrum analysis that compared activations of memorized vs. clean inputs

B.1 Shared Structure Appears Beyond Weight Decay Threshold

Following Dosovitskiy et al. [2020], we use a weight decay of 0.3 to train ViT-Base on ImageNet, but we explore how memorization structure in the weights appears as we vary this value. We find a sharp increase in disentanglement in the K-FAC basis right around this value of 0.3 (specifically, right below it at 0.27). These results are shown in Figure 5, where we also include all layers.

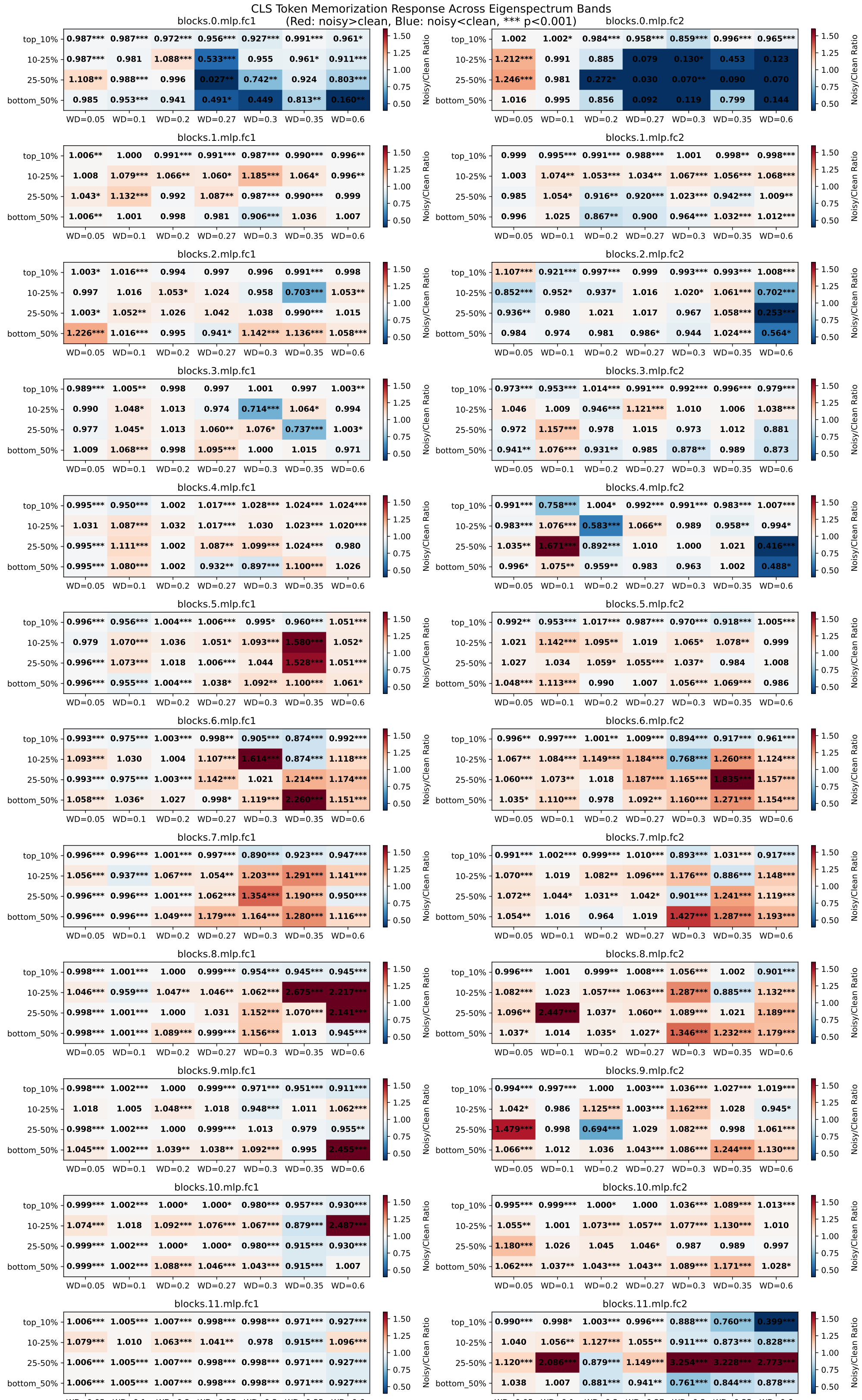

Figure 5: All MLP layers in ViT-Base trained with 10% label noise.

Prompt	Baseline	BSN	KFAC	SVD
ser.\n\nRelated Reading\n\nMore Insights\n\nCurrently we	allow the following HTML tags in comments:\n\nSingl	, 1 2 3 4 5 6 7 8 9 10 11 12 13 14	allow for two hosting options:\n\nFrom the cloud\nUs	allow for the following HTML tags in comments: \n\nS
: Use:\n & &\n <<\n >>\n [	[\n]]\n • Log In?\n\n What's my pass	1 2 3 4 5 6 7 8 9 10 11 12 13 14 1	&\n] &\n ∼∼\n # # sign\n\n • Ple	"] "\n <code>code here </code>\n\n •
prejudicial behavior of those around them.\n\nMaps &	Directories\n\nMailing Address\n\nSaint Mary's Colleg	Directories	Directories\n\n The following maps and directories a	Directions\n\n The University of Minnesota Duluth ca
7m 37s\n 5. Installing the Best Workflow color	settings\n 4m 31s\n 6. The color settings e	2 0 1 \ n 4 1 2 1 \n 2. 53 4 1 0 10 4 1 1	scheme\n 8m 26s\n 6. Choosing a color schem	scheme\n 8m 11s\n 6. Choosing t he color sch
otation\n\n Defines a table caption\n\n	Defines a citation \n\n<code>Defines computer code	(1 2 3 4 5 6 7 8 9 10 11 12 13 14	Defines a citation \n\n<code> Defines computer code	Defines a citation \n\n<code> Defines a technical la
d-0 to reset your zoom\n\nPress Ctrl-0 to reset your	zoom\n\n\nPlease upgrade Flash or install Chrome\nto	. and 10 . . and and 0 and 00 . 0	zoom\n\n\nPlease upgrade Flash to use this tool. <|en	zoom\n\n\nPlease upgrade Flash and JavaScript settin
}}\n1. {{fields.video_link. url}}\n\nReady to post!	You've uploaded the maximum number of images. \n\nYo	1, 2 = 2, 3 = 2\n\n 2. 2 1 2 2 2.com\n\n	You've uploaded the right number of images.\n\nYou'	**You've uploaded the maximum number of images.\n\nTo**
mine an appropriate range of doses for trailing ar	butus. Keep in mind that natural products are not	butus 10 20 30. 11 12 13 <|endoftext|>	butus. Keep in mind that natural products are not	**butus. Be careful when using trailing arbutus. <|en**
>Defines a short quotation\n\n<samp> Defines sample	computer code text \n\n<small> Defines small text\n\n<	1 2 3 4 1 3 2 4 5 6 7 8 9 10 11 12 13 14 15 16 17	text\n\n<small> Defines small text\n\n<span> Defines	code code\n\n <small>Defines a short quotation \n\n<sp
dows, and midtones\n 1m 16s\n 2. Introducing	the Auto commands \n 7m 23s\n 3. Adjusting C	5 2\n 4m 50 3m 21 10m 3 21m 12 7m 9	Type\n 3m 36s\n 3. Changing type size with	the Curves and Saturation\n 1m 56s\n 3. 1h

Table 3: Example generations from OLMo-2 1B from memorized prefixes. We only include the last 50 characters of the prefix for space reasons. Newlines are added for space reasons as well.

C BalancedSubnetwork Hyperparameters

D Example LM Generations

E OLMo-2-1B Pruning Results

Prompt	Baseline	BSN	KFAC	SVD
Mr. and Mrs. Dursley, of number four,	Privet Drive, were proud of the fact that they we	Privet Drive, were proud of the fact that they we	in the village of Barnsley, in the county of Bedf	Privet Drive, Little Whinging, were at home in th
The capital of France is	Paris. The French language is spoken in France. T	Paris,oppable by train from London, Paris is a cit	Paris, which is also the capital of the Île-de-Fr	Paris, which is the most important city in the co
Hello everyone, today I'm going to be	talking about the importance of the 3rd person si	talking about the importance of the 3rd person si	talking about a very important topic in the field	talking about the importance of the internet. The
Double, double toil and trouble	, Fire burn and cauldron bubble." The witches' ca	, / Cursed be his haggard eye, / And cursed	, double toil and trouble, double toil and trouble	, double, double toil and trouble, double, double
The quick brown fox jumps	over the lazy dog. The quick brown fox jumps ove	over the brown fox. The quick brown fox jumps ove	over the lazy gray dog. The quick brown fox jumps	over the brown dog. The quick brown dog jumps ove
And then, according to the plan	, the next day, the next day, the next day, the ne	, the two men would have been able to reach the su	, the next day, the next day, the next day, the ne	, the next day, the next day, the next day, the ne
And they all lived happily	ever after. The End 0 comments about this story	ever after. Well, not quite. The story of the 20	ever after. The End The End The End The End	ever after. The End The End The End The End
The northern and part of the western borders of Bloomfield Township form	the boundary o f the township with the city of Blo	the boundary of the township. The southern border	the northern and western borders of the township.	a part of the township of Bloomfield. The souther
We the People of the United States, in Order to form a more perfect Union	, establish Justice, insure domestic Tranquility,	, establish Justice, insure domestic Tranquility,	, establish Justice, insure domestic Tranquility,	, establish Justice, insure domestic Tranquility,
The opposite of up is	down. The opposite of left is right. The opposite	down, meaning that the direction of the arrow is	down. The opposite of left is right. The opposite	down. The opposite of in is out. The opposite of

Table 4: OLMo 2 1B generations highlighting random text and common but not necessarily memorized prompts. We include the prompt and then the next 50 characters generated by each odel. Newlines are added to generations to save space.

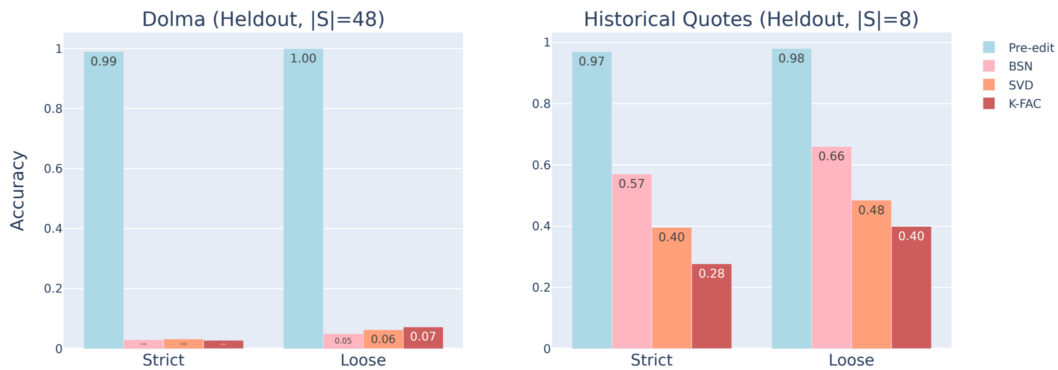

Figure 6: Editing results for BalancedSubnet, and truncating weight matrices according to SVD or K-FAC directions in OLMo2 1B.

	Memorized Data Variation Average Levenshtein Distance		Non-memorized Data Preservation nDCG@10
	Dolma	Quotes	
Pre-edit	0.01	0.01	1.00
BSN	0.90	0.25	0.91
SVD	0.78	0.40	0.91
K-FAC	0.76	0.47	0.90

Table 5: Additional statistics for OLMo 2 1B. Editing metrics showing variation from memorized suffixes (measured by Levenshtein distance) and non-memorized data's preservation of the unedited model's top 10 next token predictions. All metrics retain faithfulness to non-memorized data. BSN transfers the least to the quotes dataset, and K-FAC is the most balanced. For all metrics, higher is better.