GRADIENT CLUSTER IMBALANCE: WHY LARGE BATCHES FAVOR NORMALIZED OPTIMIZERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Adam and Muon often outperform plain stochastic gradient descent in neural network training, but their advantages vary across layers and batch sizes. Previous work has found that Adam is robust to class imbalance in the first and last layers, while Muon is effective in hidden layers. However, a unifying mechanism remains unclear, particularly how these advantages depend on batch size. We identify a property for per-sample gradients that we refer to as *gradient cluster imbalance*: per-sample gradients form clusters with similar directions that are nearly orthogonal to other clusters, and those clusters have imbalanced sizes. Under this property, we show that GD makes slow progress in directions corresponding to rare clusters for smooth, convex objectives, and that large batches offer limited gains over stochastic gradient descent as averaging leads to ill-conditioning. Under additional structural assumptions, SignGD, a simplified model of Adam, and idealized Muon mitigate this imbalance by normalizing gradients from different clusters separately. Measurements of individual layers provide partial support for these structural assumptions. Experiments reveal unequal gradient clusters across hidden layers of language models and vision models, extending existing observations of frequency imbalance beyond class labels. We show that clusters vary across layers and over training and reflect shared token patterns and visual features.

1 INTRODUCTION

Adam variants (Kingma and Ba, 2015) have been the default choice for training language models, but spectral normalization methods (Carlson et al., 2015) such as Muon (Jordan et al., 2024) have been shown to improve performance at scale (Liu et al., 2025; Shah et al., 2025; Bai et al., 2026). A common explanation invokes a low effective rank of the gradient; a few directions dominate the gradient, while many directions have small associated singular values, motivating spectral methods that normalize the update across singular vectors (Davis and Drusvyatskiy, 2026). This description, however, does not explain what property of the data or architecture gives rise to this imbalanced structure, or how it connects to the observed differences between optimizers, such as why Adam is still used in the last layer of LLMs (Jordan et al., 2024; Liu et al., 2025), or why the advantage of Adam or Muon only shows at large batch sizes (Marek et al., 2025; Srećković et al., 2025).

Frequency imbalance in language data offers a possible explanation for the source of this spectral structure. Imbalance in token frequencies translates to different gradient magnitudes for weights associated with different tokens (Kunstner et al., 2024). This mechanism is especially transparent in the last layer, where rows of the weight matrix correspond directly to output tokens, and ablation studies have found that much of Adam's benefit is in optimizing the last layer (Zhao et al., 2025; Zhang et al., 2025b). Recent work suggests that Muon's benefit in hidden layers may also be driven by frequency imbalance. Wang et al. (2026) show that Muon learns word co-occurrences faster than Adam when word-pair frequencies are imbalanced, and attribute this effect to its use in hidden layers. In hidden layers, however, frequency-associated directions are no longer aligned with individual parameter coordinates, making Adam's coordinate-wise rescaling less effective (Zhang et al., 2025a). Frequency imbalance might still exist, but without a direct correspondence between classes and specific parameter rows, the source of imbalance is harder to identify.

This leads to our central question: How can we tell whether this imbalance in frequency affects optimization beyond the last layer, and how do different optimizers interact with this structure?

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1.1 CONTRIBUTIONS.

We show evidence of what we call *gradient cluster imbalance*: per-sample gradients form clusters along similar directions that are approximately orthogonal to one another, and these clusters have imbalanced sizes, giving some directions much more weight than others in the average gradient. This observation helps explain two empirical phenomena. First, large batch sizes offer limited gains for stochastic gradient descent (SGD), because the ill-conditioning induced by gradient cluster imbalance becomes more pronounced as the batch size grows. Second, under additional assumptions, normalized updates such as Adam and Muon can outperform gradient descent by removing this imbalance.

Structure in gradients. We identify gradient cluster imbalance through block structure in per-sample gradient similarity matrices, revealing approximately orthogonal clusters of unequal sizes without requiring known class or token labels. These visualizations reveal frequency imbalance in hidden layers, extending observations previously made in the output layer of language models (Kunstner et al., 2024) and providing evidence for the hypothesis of Wang et al. (2026).

Impact on optimization. We formalize gradient cluster imbalance through a mixture model with cluster frequencies following a power law. For smooth convex objectives, we show that gradient descent needs a number of iterations t that grows with the number of clusters m to reach an error of ε when the cluster frequencies follow a power law with exponent 1 (Zipf), generalizing results of Kunstner and Bach (2025) on linear models, and showing only a logarithmic improvement of full-batch over single-sample SGD. Under additional structural assumptions, sign and spectral updates, idealized versions of Adam and Muon, can be independent of the imbalance and avoid this pitfall.

Evolution of cluster structure. We examine how the size and composition of gradient clusters vary across layers and training in language and vision models. Cluster semantics reflect the output class at initialization but evolve through training to capture more complex patterns such as bigrams in language or visual similarity in vision. Even with balanced classes, image clusters can grow imbalanced with training, reflecting shared features or classes the model confuses with one another.

2 STRUCTURE IN GRADIENTS AND FREQUENCY IMBALANCE

We begin with the known source of imbalance in the last layer of language models (Kunstner et al., 2024). The weight of the last layer is a matrix of size $[c \times d]$ where c is the number of classes and d is the hidden dimension. Due to the structure of the cross-entropy loss, the gradient for a single sample has magnitude ≈ 1 on the row corresponding to the target token, and magnitude $\approx 1/c$ on other rows.[1] When averaged, if these contributions do not cancel out, frequent tokens contribute more to their corresponding rows, and frequency imbalance leads to imbalance in the row magnitudes.

This description relies on the knowledge of the token frequencies and their relationship to the rows of the weight matrix. The question we aim to answer is how to detect whether a similar imbalance exists in other layers, when optimizing a loss function ℓ over a data distribution $p(z)$ with weights $\mathbf{w}$, $\mathcal{L}(\mathbf{w}) = \mathbb{E}_{z\sim p(z)}[\ell(\mathbf{w}; z)]$. We propose a generalization of the above structure based on mixtures of almost-orthogonal per-sample gradients, similar to the mixture model of Feldman (2020).

Definition 1 (Orthogonal Gradient Clusters)**.** *Suppose our data z comes from a distribution $p(z)$, which is a mixture of m clusters $\mathcal{C}_1, ..., \mathcal{C}_m$, where each component has a different probability $\pi_1, ..., \pi_m$. We assume that the per-sample gradients are aligned if they come from the same component, and nearly orthogonal otherwise. For $0 < \varepsilon < \delta$ and any $z \in \mathcal{C}_i, z' \in \mathcal{C}_j$,*

$$|\cos(\nabla\ell(\mathbf{w}; z), \nabla\ell(\mathbf{w}; z'))| \leq \varepsilon \quad \text{if } i \neq j, \qquad |\cos(\nabla\ell(\mathbf{w}; z), \nabla\ell(\mathbf{w}; z'))| \geq \delta \quad \text{if } i = j.$$

This definition captures the idea that similar samples have similar gradients (Chatterjee, 2020), but are almost orthogonal to gradients of samples from other, dissimilar clusters. We say a problem suffers from **Gradient Cluster Imbalance** if it exhibits orthogonal gradient clusters with imbalanced frequencies $\pi_1, ..., \pi_m$, as is the case in the last layer of classification problems. In such cases, directions in parameter space associated with samples from frequent clusters receive larger updates than directions associated with rare clusters, leading to imbalanced updates.

[1] The gradient of the cross-entropy loss with respect to the logits is $\mathbf{p} - \mathbf{e}_y$, where $\mathbf{p}$ is the vector of predicted probabilities and $\mathbf{e}_y$ is the one-hot encoding of the target class. At initialization, $\mathbf{p}$ is uniform and the entry corresponding to the target token has magnitude $1 - 1/c \approx 1$, while each remaining entry has magnitude $1/c$.

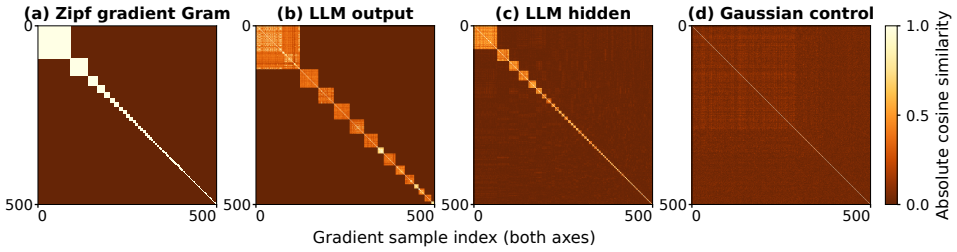

Figure 1: **Gradient-similarity blocks reveal clusters of directions and frequency imbalance.** Matrix of absolute cosine similarities between per-sample gradients, after reordering using DBSCAN. (a) Synthetic examples with one-hot vectors, frequencies following $\pi_k \propto 1/k$ (Zipf's law). (b–c) Per-sample gradients for a language model, output layer and hidden layer at initialization. (d) Comparison on Gaussian samples matching the mean and covariance of the hidden-layer gradients in (c).

2.1 Visualizing the gradient structure and frequency imbalance

We describe a method to visualize the structure of per-sample gradients and check its consistency with Definition 1. For a batch of n samples, we compute the gradients for each sample with respect to some subset of weights, $\mathbf{g}_1, \ldots, \mathbf{g}_n$, and compute the $\mathbf{S} \in \mathbb{R}^{n \times n}$ matrix of absolute cosine similarities,

$$\mathbf{S}_{ij} = |\cos(\mathbf{g}_i, \mathbf{g}_j)|.$$

Approximately orthogonal gradient clusters produce small between-group entries in $\mathbf{S}$ and larger within-group entries. We look for this structure by clustering samples using absolute cosine similarity and permuting the rows and columns to reveal a block-diagonal structure. Figure 1 illustrates this process. Panel (a) shows the absolute cosine similarities between one-hot vectors $\mathbf{g}_i = \mathbf{e}_{c_i}$, where $\mathbb{P}(c_i = k) \propto 1/k$ for $k \in [150]$. The similarity is either 0 or 1, producing a block-diagonal structure with blocks of size $\propto 1/k$. The same structure appears with per-sample gradients w.r.t. the weight of the last layer of a language model at initialization (b), but also in a hidden layer (c). Although similarities are not $\{0, 1\}$, the matrix shows structured blocks of unequal sizes. To verify that the cluster structure is not an artifact of the clustering method but a real property of the data, we apply the same methodology to samples from a Gaussian distribution with the same mean and covariance, shown in panel (d) and Appendix D. We do not observe the pronounced blocks seen in real gradients, indicating that the per-sample gradients are more structured than Gaussian noise.

Technical details. The language model used is Pythia-1B (Biderman et al., 2023). We use DBSCAN (Ester et al., 1996) to group and reorder gradients, a standard density-based clustering method. We give additional details on the models and datasets in Appendix B, the clustering method in Appendix C, and show that the results can be qualitatively reproduced with other clustering or seriation methods in Appendix C.5.

Limitations. We note that such clean clustering does not appear at every layer. The cluster structure changes with network depth and during training. We investigate their evolution and semantic meaning in Section 4, but first look at how this structure affects optimization dynamics.

3 Gradient Cluster Imbalance and Optimization Dynamics

In this section, we discuss the implications of gradient cluster imbalance on optimization, showing that it can replicate behaviors observed in practice: the poor performance of gradient descent on heavy-tailed data, its poor scaling with batch size, and the benefits of sign and spectral descent methods under additional structural assumptions. Throughout this section, we use the following simplified problem as an illustrative example to show the effects of gradient cluster imbalance. To discuss functions that can be represented by a vector $\mathbf{w}$ or a matrix $\mathbf{W}$, we use the convention $\mathbf{w} = \mathrm{vec}(\mathbf{W})$.

Example 1. *Given m pairs of mutually orthonormal vectors $(\mathbf{u}_i, \mathbf{v}_i)$, such that $\langle \mathbf{u}_i, \mathbf{u}_j \rangle = 0$ and $\langle \mathbf{v}_i, \mathbf{v}_j \rangle = 0$ for $i \neq j$, and $\|\mathbf{u}_i\| = \|\mathbf{v}_i\| = 1$ sampled with replacement with probabilities $\pi_1, ..., \pi_m$ following a power-law $\pi_i \propto i^{-\alpha}$ for $\alpha \geq 0$. We learn a linear model that maps inputs $(\mathbf{u}_i, \mathbf{v}_i)$ to 1.*

$$\mathcal{L}(\mathbf{W}) = \frac{1}{2} \sum_{i=1}^{m} \pi_i (\mathbf{u}_i^\top \mathbf{W} \mathbf{v}_i - 1)^2.$$

The example is inspired by associative memory models (e.g., Cabannes et al., 2024; Wang et al., 2026) as the weights $\mathbf{W}$ end up learning pairwise associations between the vectors $\mathbf{u}_i$ and $\mathbf{v}_i$. While not a realistic model, it is sufficient to observe the effects of gradient cluster imbalance. The gradient for sample i is in the direction $\mathbf{u}_i\mathbf{v}_i^\top$, orthogonal to the gradients of other samples, and their frequencies determine their weights in the average gradient. As such, it acts as a simple example of a problem satisfying the more general conditions we will introduce over the next sections, starting with smooth, convex functions. We refer to classical optimization settings such as smoothness and convexity as *conditions*, and to the structural properties of the gradients introduced in this work as *assumptions*.

Condition 1. *We assume the objective function $\mathbf{w} \mapsto \ell(\mathbf{w}; z)$ is convex and L-smooth in $\mathbf{w}$ for all z,*

$$0 \leq \ell(\mathbf{w}'; z) - \ell(\mathbf{w}; z) - \langle \nabla \ell(\mathbf{w}; z), \mathbf{w}' - \mathbf{w} \rangle \leq \frac{L}{2} \|\mathbf{w}' - \mathbf{w}\|^2 \quad \textit{for all} \quad z, \mathbf{w}, \mathbf{w}'.$$

We start with this well-known problem class to illustrate the effect of gradient cluster imbalance on optimization; the same approach could be pursued in non-convex or non-smooth settings. To model the impact of gradient cluster imbalance, we make a stronger assumption than Definition 1.

Assumption 1 (Strong separation)**.** *The data distribution is a mixture of m clusters, $p(z) = \sum_{k=1}^m \pi_k p_k(z)$, where p_k has supports on $\mathcal{C}_k$, $\pi_k \propto 1/k^\alpha$ for $\alpha > 0$. The per-sample gradients are orthogonal across clusters and parameters;*

$$\langle \nabla \ell(\mathbf{w}; z), \nabla \ell(\mathbf{w}'; z') \rangle = 0 \quad \textit{for all} \quad z \in \mathcal{C}_i,\, z' \in \mathcal{C}_j,\, i \neq j, \quad \textit{and for all} \quad \mathbf{w}, \mathbf{w}'.$$

Assumption 1 is stronger than Definition 1; it requires exact orthogonality rather than a tolerance of ε, and requires that the orthogonality holds globally rather than at a single $\mathbf{w}$. This effectively separates the problem into m independent subproblems in different directions. While too strong to be realistic, this assumption isolates the effect of frequency imbalance. Allowing approximate orthogonality and slowly varying component subspaces rather than global orthogonality would better reflect the observed structure, but would introduce interactions between clusters into the analysis, obscuring the simple intuition we want to convey. Finally, we assume that the clusters are otherwise similar.

Assumption 2. *The clusters are similar under ℓ at $\mathbf{w}_0$, in that the following bounds apply for the expected loss on each component k, $\mathcal{L}_k(\mathbf{w}) = \mathbb{E}_{z \sim p_k(z)}[\ell(\mathbf{w}; z)]$, with $R, \Delta_{\min}, \Delta_{\max} > 0$,*

$$\left\|\mathbf{w}_0 - \mathbf{w}^{k*}\right\| \leq R, \quad \textit{and} \quad \mathcal{L}_k(\mathbf{w}_0) - \mathcal{L}_k(\mathbf{w}^{k*}) \in [\Delta_{\min}, \Delta_{\max}], \quad \textit{where} \quad \mathbf{w}^{k*} = \arg\min_{\mathbf{w}} \mathcal{L}_k(\mathbf{w}).$$

3.1 Impact on gradient descent

Using those assumptions, we can recover known behavior of gradient descent on imbalanced data. We start with the impact of the imbalance, showing that the convergence of gradient descent is slowest when the data follows a power law with $\alpha = 1$, recovering similar results to those obtained on a linear bigram model by Kunstner and Bach (2025), but holding for arbitrarily smooth, convex problems. We illustrate this effect on Example 1 in Figure 2, and formalize it in the next theorem. The main intuition behind the result, derived in Appendix A.1, is that under Assumption 1, running gradient descent with step-size η on the average problem $\mathcal{L}$ is equivalent to running gradient descent independently on each component $\mathcal{L}_k$ with step-size $\eta\pi_k$. The most frequent component limits the shared step-size, so rare clusters receive smaller effective steps, leading to slow convergence in the presence of imbalance.

Theorem 1. *Under Condition 1 and Assumptions 1 and 2, gradient descent with step-size $\eta = 1/(\pi_1 L)$ guarantees a relative error of $\varepsilon \in (0, 1)$ within T iterations, where*

$$T = O\left(\begin{cases} m^\alpha & \textit{if } 0 \leq \alpha < 1 \\ m^{1 - c_1 \varepsilon} & \textit{if } \alpha = 1 \\ 1 & \textit{if } \alpha > 1 \end{cases}\right), \quad \textit{with} \quad c_1 := \frac{\Delta_{\min}}{\Delta_{\max}} \in (0, 1].$$

Here, big-O is asymptotic in terms of m and hides terms depending on $L, R, \Delta, \varepsilon$.

Proof sketch. Applying standard smooth-convex bounds to each component with step-size $\eta\pi_i$,

$$\frac{\mathcal{L}(\mathbf{w}_T) - \mathcal{L}^*}{\mathcal{L}(\mathbf{w}_0) - \mathcal{L}^*} \leq \sum_{k=1}^m \pi_k \min\left\{1,\ \frac{1}{T\eta\pi_k(2 - L\eta\pi_k)} \frac{2R^2}{\Delta_{\min}}\right\}.$$

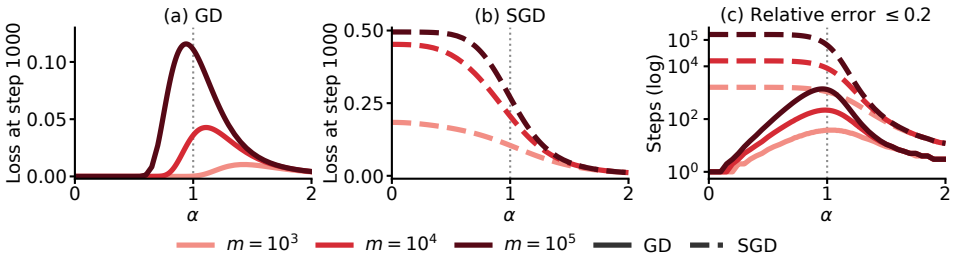

Figure 2: **Gradient descent is slowed down by rare clusters.** Performance of gradient descent (GD) and stochastic gradient descent (SGD) with batch size 1 on Example 1 for different values of α and m. As predicted by Theorems 1 and 2, the performance of GD is worst at $\alpha = 1$ (a) while the performance of SGD is worst at $\alpha = 0$ (b). The improvement of going from SGD to GD is worst near $\alpha = 1$, showing the difficulty in increasing parallelism under imbalance (c).

The step-size is limited by the most frequent component, as $\eta < (2/L\pi_1)$ for stability, but does not make progress on rare clusters when $T\eta\pi_k \ll 1$. With $\eta = 1/(L\pi_1)$, the error can be simplified to

$$\frac{\mathcal{L}(\mathbf{w}_T) - \mathcal{L}^*}{\mathcal{L}(\mathbf{w}_0) - \mathcal{L}^*} \leq \frac{2LR^2k}{T\eta\Delta_{\min}H_{m,\alpha}} + \frac{H_{m,\alpha} - H_{k,\alpha}}{H_{m,\alpha}}, \quad \text{for any } k \in [m],$$

where $H_{a,b} = \sum_{i=1}^{a} i^{-b}$ is the generalized harmonic number. The first term bounds the remaining error on the k most frequent clusters, while the second bounds the tail by its initial error. Bounding those terms and optimizing over k gives the stated bounds for large m. □

This result generalizes the phase transition depending on α of Kunstner and Bach (2025) for a linear model with quadratic loss. For $\alpha > 1$, the tail of the distribution is so light that rare clusters do not contribute meaningfully to the total error. For $\alpha < 1$, the distribution is more uniform, and the impact of rare clusters is less pronounced. The case $\alpha = 1$ has both large imbalance and a heavy tail, leading to the worst scaling of gradient descent with the number of clusters m if ε is small.

Stochastic gradient descent. The performance of SGD on LLMs is very dependent on the batch size; gradient descent performs poorly at large batch sizes compared to Adam (Kunstner et al., 2023), but is competitive at small batch sizes (Marek et al., 2025; Srećković et al., 2025). As illustrated in Figure 2, the poor performance of full-batch GD versus single-sample SGD is most pronounced under gradient cluster imbalance when $\alpha = 1$, and intermediate batch sizes interpolate between these extremes. To simplify the proof for SGD, we look at problems satisfying interpolation (Vaswani et al., 2019).

Condition 2 (Interpolation)**.** *The loss is minimized at* $\mathbf{w}^*$ *for all clusters and all samples*

$$\nabla\ell(\mathbf{w}^*; z) = 0 \quad \textit{for all } z.$$

Typical proofs in the stochastic regime have to balance a bias-variance tradeoff, reducing the step-size or increasing the batch size as we get closer to the minimum to reduce the error due to sampling. Under interpolation, a constant step-size works, allowing for a more direct comparison with gradient descent. Although this setting is likely overly optimistic for the performance of gradient descent with a small batch size, it lets us focus on the main difficulty of the problem, that we see rare clusters infrequently when running stochastic gradient descent.

Theorem 2. *Under Assumptions 1 and 2 and Conditions 1 and 2, SGD with step size* $1/L$ *reaches expected relative population error* ε *at its averaged iterate within*

$$T = O\left(\begin{cases} m & \text{if } 0 \leq \alpha < 1 \\ m^{1-c_2\varepsilon}\log m & \text{if } \alpha = 1 \\ 1 & \text{if } \alpha > 1 \end{cases}\right), \qquad \text{with} \qquad c_2 := \frac{2\Delta_{\min}}{LR^2} \in (0, 1].$$

Here, big-O is asymptotic in terms of m *and hides terms depending on* $L, R, \Delta, \varepsilon$.

The proof follows a similar idea as for Theorem 1, and is given in Appendix A.2. The greatest difference between the stochastic and deterministic rates is that the performance of stochastic gradient descent is worst when $\alpha \to 0$. This difference arises because of the step-size. For stochastic gradient descent with batch size 1, the step-size is $\eta \propto 1/L$, while the maximum stable step-size that satisfies the stability requirement is $\eta \propto 1/(\pi_1 L)$. In the uniform case where $\pi_k = 1/m$, gradient descent

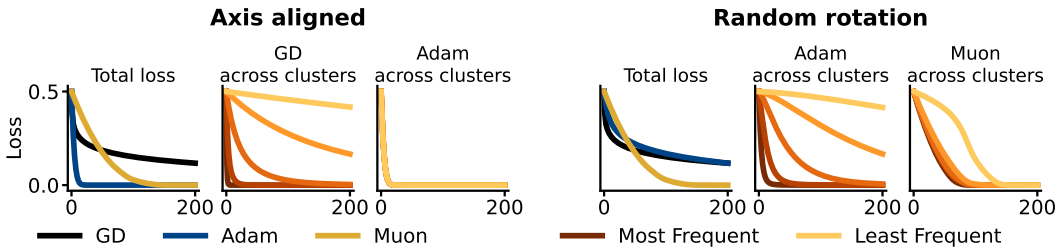

Figure 3: **Whether sign- and spectral descent help depends on axis- and spectral alignment.** Overall performance and decomposition of the loss across clusters ordered by frequency, to observe the effect of imbalance on training on two variants of Example 1. **Left:** Under axis-aligned gradient cluster imbalance (Assumption 3), GD is slow on rare clusters while Adam makes fast progress across clusters. **Right:** Rotation removes axis alignment, making Adam slow on rare clusters. Muon remains fast as the problem is spectrally-aligned (Assumption 4). Its slow convergence on the least frequent clusters is due to imperfect normalization using Newton–Schulz, see Appendix B.1.

can use a large step-size $\eta = m/L$ and reduce the number of iterations by a factor of m. SGD might still be preferable as each iteration is much cheaper, but at least there is a clear advantage to GD in terms of per-iteration progress as it effectively takes m steps in parallel across all clusters, which justifies its higher per-iteration cost of requiring more samples. The maximum stable step-size is smaller for larger α, and at $\alpha = 1$, the improvement of GD over SGD is only logarithmic in m, even though GD needs $\gg m$ more samples per iteration than SGD. This gives additional context for the difficulty of training with clustered data; in the stochastic setting, the problem is hardest when the distribution is uniform, but this case parallelizes easily by using a large step-size. In contrast, $\alpha = 1$ is the worst case for deterministic training, as averaging per-sample gradients introduces ill-conditioning by giving infrequent clusters a small effective step-size. This argument complements existing work on diminishing returns from increasing batch size (**shallue2019measuring**) and their relationship to gradient diversity (Yin et al., 2018), by identifying imbalanced cluster frequencies as a mechanism that limits the benefits of large batches.

3.2 Impact on sign and spectral methods

The previous section looked at how imbalanced orthogonal gradient clusters slows down gradient descent, and how this effect is most pronounced in the deterministic setting under power-law cluster frequencies with exponent $\alpha = 1$. We now turn our attention to how sign and spectral methods can mitigate the problem. Sign and spectral methods differ from standard gradient descent by normalizing the gradient. Sign methods normalize coordinate-wise, while spectral methods normalize the singular values of the gradient matrix. To discuss such updates, we introduce the matrix form of the parameters $\mathbf{W} \in \mathbb{R}^{d_1 \times d_2}$, assuming that the vector form we used so far is the vectorization of this matrix, $\mathbf{w} = \mathrm{vec}(\mathbf{W})$. The sign and spectral updates we discuss are

$$\text{Sign descent:} \qquad \mathbf{w}_{t+1} = \mathbf{w}_t - \eta\, \mathrm{sign}(\nabla \mathcal{L}(\mathbf{w}_t)),$$

$$\text{Spectral descent:} \qquad \mathbf{W}_{t+1} = \mathbf{W}_t - \eta \mathbf{U}_t \mathbf{V}_t^\top, \quad \text{where } \nabla \mathcal{L}(\mathbf{W}_t) = \mathbf{U}_t \mathbf{\Sigma}_t \mathbf{V}_t^\top,$$

Adam and Muon are variants of such updates that include momentum for smoothing and, in the case of Muon, an approximate spectral update using Newton-Schulz.

The normalization operation is often thought to be beneficial, correcting for an imbalance that should not be present in the first place. But it is not a priori obvious why normalization helps; after all, these methods ignore information about the relative magnitude of the gradients. However, if the imbalance in coordinates or singular values were due to the relative frequency of the clusters in Assumption 1, which makes gradient descent slow, normalization could remove this effect and accelerate convergence by implicitly using a different step-size for each component. We show that this is indeed the case, under additional assumptions on the structure that connect the gradient clusters to the coordinates or singular values of the gradient matrix. Sign methods are robust to imbalanced gradient clusters if they are axis-aligned, while spectral methods are robust to imbalanced gradient clusters if they are spectrally-aligned, which we will now define.

Sign methods and axis-aligned gradient clusters. Sign methods normalize the gradient coordinate-wise, and therefore need a form of axis-alignment to be effective; they are not robust to arbitrary rotations of the data (Zhang et al., 2025a). To illustrate this effect, we consider Example 1 with different data distributions, shown in Figure 3. If the data $(\mathbf{u}_i, \mathbf{v}_i)$ are one-hot vectors, the imbalance is axis-aligned; the difference in the component frequencies translates to a difference in gradient magnitude across coordinates, and sign-like methods can outperform gradient descent. However, if we rotate the data $(\mathbf{u}_i, \mathbf{v}_i)$ by a random rotation matrix, the axis-alignment is broken, and Adam no longer outperforms gradient descent. To explain this behavior, we introduce additional structural assumptions on the per-sample gradients.

Assumption 3 (Axis-aligned gradient clusters)**.** *We say orthogonal gradient clusters (Assumption 1) are axis-aligned if the coordinates of the problem can be partitioned into m disjoint sets $\mathcal{I}_1, \dots, \mathcal{I}_m$ such that per-sample gradients from component k are contained in the span of the coordinates $\mathcal{I}_k$,*

$$\nabla \ell(\mathbf{w}; z) \in \operatorname{span}(\mathcal{I}_k) \quad \textit{for all } z \in \mathcal{C}_k.$$

This assumption is satisfied by Example 1 if the vectors $(\mathbf{u}_i, \mathbf{v}_i)$ are one-hot vectors, and is approximately satisfied in the last layer of language models. Under this assumption, the sign operation separates over the clusters of the problems, canceling the effect of the frequency weights;

$$\operatorname{sign}(\nabla \mathcal{L}(\mathbf{w})) = \operatorname{sign}(\textstyle\sum_{k=1}^m \pi_k \nabla \mathcal{L}_k(\mathbf{w})) = \sum_{k=1}^m \operatorname{sign}(\nabla \mathcal{L}_k(\mathbf{w})).$$

As a result, sign descent on the average loss $\mathcal{L}$ with step-size η is equivalent to running sign descent on each component $\mathcal{L}_k$ independently, with step-size η (instead of $\eta\pi_k$ for gradient descent).

Spectral methods and spectrally aligned gradient clusters. A similar behavior can be observed for spectral methods, under a different structure for the orthogonal gradient clusters.

Assumption 4 (Spectrally-aligned gradient clusters)**.** *We say orthogonal gradient clusters (Assumption 1) are spectrally aligned if the per-sample gradient matrices from different clusters are row- and column-orthogonal, that is, for $z \in \mathcal{C}_i$ and $z' \in \mathcal{C}_j$, $i \neq j$,*

$$\nabla \ell(\mathbf{W}; z)^\top \nabla \ell(\mathbf{W}'; z') = 0, \quad \textit{and} \quad \nabla \ell(\mathbf{W}; z) \nabla \ell(\mathbf{W}'; z')^\top = 0, \quad \textit{for all} \quad \mathbf{W}, \mathbf{W}' \in \mathbb{R}^{d_1 \times d_2}.$$

First, note that Assumption 4 reduces to Assumption 1 when $\mathbf{W}$ is a vector, i.e. when $d_1 = 1$ or $d_2 = 1$. When $\mathbf{W}$ is a matrix, Assumption 4 implies Assumption 1, as $A^T B = 0$, and $AB^T = 0$ implies that $\langle \operatorname{vec}(A), \operatorname{vec}(B) \rangle = 0$. The converse does not hold: suppose A and B are rank-1 matrices, as is the case for gradients of linear layers, which are outer products of the input and a backpropagated vector. If $A = \mathbf{u} \otimes \mathbf{x}$ and $B = \mathbf{v} \otimes \mathbf{y}$, $A^\top B = AB^\top = 0$ means that $\langle \mathbf{u}, \mathbf{v} \rangle = 0$ and $\langle \mathbf{x}, \mathbf{y} \rangle = 0$, while $\langle A, B \rangle = 0$ only requires one inner product to be zero. What this assumption implies is that the orthogonal gradient clusters separate along the singular vectors of the gradient matrices, that is,

$$\nabla \mathcal{L}(\mathbf{W}) = U \Sigma V^\top \quad \text{with} \quad \begin{aligned} U &= [U_1, \dots, U_m], \\ \Sigma &= \operatorname{BlockDiag}(\Sigma_1, \dots, \Sigma_m), \\ V &= [V_1, \dots, V_m], \end{aligned} \quad \text{where} \quad \nabla \mathcal{L}_k(\mathbf{W}) = U_k \Sigma_k V_k^\top.$$

Assumption 4 is the equivalent of Assumption 3 for spectral instead of sign descent; Spectral descent on the average loss $\mathcal{L}$ with step-size η is equivalent to spectral descent on each component $\mathcal{L}_k$ independently, with step-size η. In both cases, assuming that the clusters are otherwise similar, the performance of sign or spectral descent under their respective structural assumption does not depend on the frequency weights π_k or the number of clusters m.

These results are similar to the scale-freeness argument of Zhuang et al. (2022, Thm 3.1), showing that the performance of Adam on diagonal problems is independent of diagonal rescaling. While they do not give an explicit rate, they show that performance does not suffer from the same slowdown due to rare clusters as gradient descent with large batches. Gradient cluster imbalance provides a mechanism for why optimization in non-Euclidean geometries helps, complementing works that study optimization in ℓ_∞-norm (Balles et al., 2020), or spectral methods (Davis and Drusvyatskiy, 2026; Ma et al., 2026), generalizing the imbalance observation of Kunstner et al. (2024).

Evidence of structured gradient cluster imbalance. Assumptions 3 and 4 are strong since they require exact orthogonality, but relaxed versions of these assumptions may be satisfied in practice. In Appendix E, we show that the violation of these assumptions is small, in the last layer for axis-alignment and in hidden layers for spectral-alignment. An observation of potential interest to the

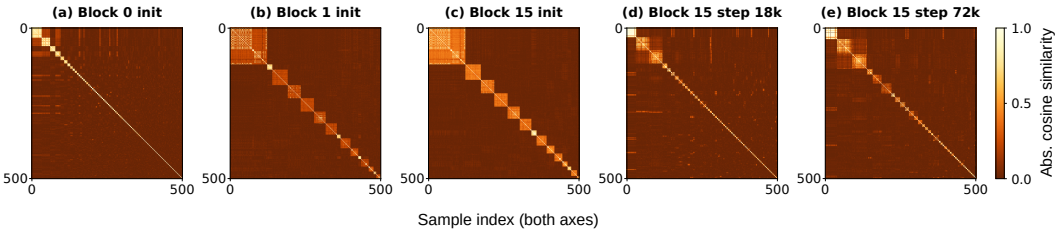

Figure 4: **Gradient clusters vary with depth and training.** As in fig. 1, absolute cosine similarity matrix between per-sample gradients for MLP blocks in Pythia-1B. **Left (a, b, c):** Variations across depth in blocks 0, 1, 15. Clusters are primarily influenced by output class, except in very early layers where we see smaller clusters, which depend on both input and output tokens. **Right (c, d, e):** Variations across training time in block 15. Clusters are initially large, primarily determined by output class, but become more refined as training progresses and the network distinguishes patterns.

design of left- and right-preconditioned methods such as Shampoo and Soap (Gupta et al., 2018; Vyas et al., 2025) is that column-orthogonality appears to be stronger than row-orthogonality. Additional results across layers and training checkpoints are provided in Appendices E.1 and E.2.

4 Evolution of Gradient Clusters and Semantic Meaning

We examine gradient clusters in language and vision models, and how these clusters vary across layers and training. For language models, we study a small 2-layer transformer on WikiText (Merity et al., 2017), as well as Pythia-1B (Biderman et al., 2023) on the Pile (Gao et al., 2020). For vision, we study ResNet-20 (He et al., 2016) on CIFAR-100 (Krizhevsky, 2009). Experimental details are provided in Appendix B.2 and Appendix B.3.

Language tasks. Both language models show clusters of unequal size in the output layer and hidden layers, although the strength of this structure varies. We observe some general trends, illustrated for Pythia-1B MLP down-projections in Figure 4 and shown in more details in Appendices F.1 and F.2; clusters are typically largest in layers closer to the output (a–c), and earlier in training (d–e). This structure and its changes through depth and training is seen across most layer types, except for gradients w.r.t. query and key projection parameters. Those layers show no clustering at initialization, while clusters appear in blocks close to the output after training, compare Figures 25 and 26 in Appendix F.2. This observation is consistent with the smaller gains from Muon on query and key matrices reported by Wang et al. (2026).

Vision tasks. In ResNet-20, at initialization, the clusters in deeper layers are balanced, reflecting the balanced classes in CIFAR-100, while gradients in early layers are all heavily correlated with each other. As training progresses, later layers show imbalanced clusters, suggesting that similarity between samples evolves as the network learns features. See details in Appendix G.1.

4.1 Semantics of the clusters

We inspect samples within gradient clusters to understand what they have in common. In both language and vision models, the examples reveal shared patterns that extend beyond class labels.

Cluster patterns in language model clusters. At initialization, the cluster patterns are primarily determined by the output class in the deep layers, and also by the last token of the input sequence in early layers, giving smaller cluster sizes. Later in training, the clusters can correspond to more complex patterns, such as words or phrases related to specific topics or linguistic constructions. Table 1 shows samples from the three largest clusters in the output layer of Pythia-1B during training, which correspond to groups of similar punctuation or stop words, see Appendix F.3 for additional examples.

Cluster patterns in vision model clusters. The size of gradient clusters is uniform at the start of training, and each corresponds to a class label, in deep layers of the network. Early layers show no clear structure. Layer in training, clusters in deep layers become imbalanced and contain images that are easily confused with each other due to visual similarity (e.g., apple/bowl/peppers, bus/streetcar/train, or dolphin/shark/whale), consistent with prior observations that pairwise gradient similarity reflect superclass structures (Fort et al., 2020). See Figure 5 and Appendix G.2 for additional examples.

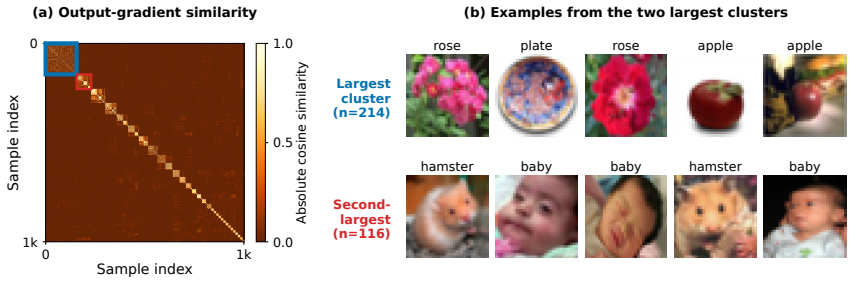

Figure 5: **In vision, large clusters contain visually similar examples. Left:** HDBSCAN clustering results on the ResNet-20 output layer at epoch 8 of CIFAR-100 training, using the absolute cosine similarity between per-sample gradients with the two largest clusters highlighted. **Right:** Five samples from each cluster showing visual similarities.

Table 1: **In language, large gradient clusters share common output tokens.** Examples from the three largest gradient clusters in Pythia-1B in the output layer at step 36,000. Cluster sizes are in percentages of all 2,048 analyzed samples. Bold text marks the target token; examples are the first three cluster members in original sample order.

Shared pattern	Size (%)	Samples (Context → target)
Line breaks	6.79	… Related Topics Catalonia independence protests → **[newline]** … the time for dialogue," said Carles Puigdemont → **[newline]** … won a slim majority in a regional election. [newline] → **[newline]**
Punctuation	6.15	… Puigdemont [newline] [newline] Catalonia's ousted leader → **,** … separatist parties won a slim majority in a regional election → **.** … is living in self-imposed exile, or another EU country → **.**
"the" / "for"	2.69	Catalonia election: Puigdemont calls → **for** "Now is → **the** "Now is the time → **for**

5 Discussion

We introduce gradient cluster imbalance to describe unequal frequencies of gradient signals beyond class labels and coordinate-aligned directions. Our analysis shows how this imbalance slows GD and limits the benefits of batching under power-law with exponent $\alpha = 1$ (Zipf's law). Under their respective structural assumptions, SignGD and Muon cancel frequency weights by normalizing component gradients independently. These results help explain Adam's advantage under axis-aligned imbalance and Muon's ability to handle certain forms of imbalance in hidden layers.

The orthogonal gradient clusters assumption is similar in spirit to other works using similarities between per-sample gradients to select data for training, for example for online subsampling (Faghri et al., 2020) or as a coreset selection method (Mohanty et al., 2025). However, the interaction between specific optimizers and the structure of per-sample gradients is underexplored. Closest to our work might be the observation of Rosenfeld and Risteski (2024), who show that instabilities in training can arise from opposite signals from similar samples, which would correspond to large clusters in our setting. The orthogonal gradient clusters structure is consistent with the idea that a model fits a mixture of different subtasks, explored in the context of scaling laws by Michaud et al. (2023) and Nam et al. (2024). Our work provides a connection to the optimization dynamics.

Our experiments reveal unequal gradient clusters in both language and vision models, with shared token patterns and visual features. While individual layers do not exactly satisfy our structural assumptions, they show strong structure suggesting that the assumptions are reasonable approximations. Together, these findings support gradient cluster imbalance as a mechanism connecting gradient structure and optimizer performance. Future work remains to develop a more realistic convergence theory under relaxed assumptions, and how this structure can be leveraged to improve optimizers.

AI USE STATEMENT

We used generative AI tools to assist with the implementation of the experiments and figures, and for reviewing the manuscript and organizing mathematical claims. We have not used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, support qualitative and thematic data analysis, interpret results, and generate synthetic data sets. Assisting with translation, cleaning, and reformatting datasets does not apply to this work. Additionally, we used generative AI tools to create or edit software code, brainstorm, format references, suggest a structure for a research paper, and propose a title or keywords for a research paper. We have reviewed AI-assisted work. Each proof and experimental result was reviewed by at least two authors. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The theoretical assumptions and results are stated in the main text, with proofs provided in Appendix A. Appendix B describes all experimental details for the synthetic example, both language models and a vision model, including architecture, datasets, optimizers, hyperparameters, and gradient extraction. Appendix C describes clustering methods and visualization procedures.

REFERENCES

Bai, Yifan, Yiping Bao, Y. Charles, et al. (2026). *Kimi K2: Open Agentic Intelligence*. arXiv preprint arXiv:2507.20534.

Balles, Lukas, Fabian Pedregosa, and Nicolas Le Roux (2020). *The Geometry of Sign Gradient Descent*.

Bar-Joseph, Ziv, David K. Gifford, and Tommi S. Jaakkola (2001). "Fast optimal leaf ordering for hierarchical clustering". In: *Bioinformatics* 17, S22–S29.

Biderman, Stella, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. (2023). "Pythia: A suite for analyzing large language models across training and scaling". In: *International Conference on Machine Learning (ICML)*.

Cabannes, Vivien, Berfin Simsek, and Alberto Bietti (2024). "Learning Associative Memories with Gradient Descent". In: *International Conference on Machine Learning (ICML)*.

Campello, Ricardo J. G. B., Davoud Moulavi, and Jörg Sander (2013). "Density-Based Clustering Based on Hierarchical Density Estimates". In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*.

Carlson, David E, Edo Collins, Ya-Ping Hsieh, Lawrence Carin, and Volkan Cevher (2015). "Preconditioned spectral descent for deep learning". In: *Conference on Neural Information Processing Systems (NeurIPS)*.

Chatterjee, Satrajit (2020). "Coherent Gradients: An Approach to Understanding Generalization in Gradient Descent-based Optimization". In: *International Conference on Learning Representations (ICLR)*.

Davis, Damek and Dmitriy Drusvyatskiy (2026). *When do spectral gradient updates help in deep learning?* arXiv preprint arXiv:2512.04299.

Ester, Martin, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu (1996). "A density-based algorithm for discovering clusters in large spatial databases with noise". In: *Knowledge Discovery in Databases (KDD)*.

Faghri, Fartash, David Duvenaud, David J. Fleet, and Jimmy Ba (2020). *A Study of Gradient Variance in Deep Learning*. arXiv preprint arXiv:2007.04532.

Feldman, Vitaly (2020). "Does Learning Require Memorization? A Short Tale about a Long Tail". In: *ACM Symposium on Theory of Computing (STOC)*, pp. 954–959.

Fort, Stanislav, Paweł Krzysztof Nowak, Stanislaw Jastrzebski, and Srini Narayanan (2020). *Stiffness: A New Perspective on Generalization in Neural Networks*.

Gao, Leo, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. (2020). *The Pile: An 800gb dataset of diverse text for language modeling*. arXiv preprint arXiv:2101.00027.

Gupta, Vineet, Tomer Koren, and Yoram Singer (2018). "Shampoo: Preconditioned Stochastic Tensor Optimization". In: *International Conference on Machine Learning (ICML)*. Vol. 80, pp. 1837–1845.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep Residual Learning for Image Recognition". In: *Computer Vision and Pattern Recognition Conference (CVPR)*.

Jordan, Keller, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein (2024). *Muon: An Optimizer for Hidden Layers in Neural Networks*. `https://kellerjordan.github.io/posts/muon/`.

Kingma, Diederik P. and Jimmy Ba (2015). "Adam: A method for stochastic optimization". In: *International Conference on Learning Representations (ICLR)*.

Krizhevsky, Alex (2009). *Learning Multiple Layers of Features from Tiny Images*. Tech. rep. University of Toronto.

Kunstner, Frederik and Francis R. Bach (2025). "Scaling laws for gradient descent and sign descent for linear bigram models under Zipf's law". In: *Conference on Neural Information Processing Systems (NeurIPS)*.

Kunstner, Frederik, Jacques Chen, Jonathan Wilder Lavington, and Mark Schmidt (2023). "Noise Is Not the Main Factor Behind the Gap Between SGD and Adam on Transformers, But Sign Descent Might Be". In: *International Conference on Learning Representations (ICLR)*.

Kunstner, Frederik, Alan Milligan, Robin Yadav, Mark Schmidt, and Alberto Bietti (2024). "Heavy-tailed class imbalance and why Adam outperforms gradient descent on language models". In: *Conference on Neural Information Processing Systems (NeurIPS)*.

Liu, Jingyuan, Jianlin Su, Xingcheng Yao, et al. (2025). *Muon is scalable for LLM training*. arXiv preprint arXiv:2502.16982.

Ma, Jianhao, Yu Huang, Yuejie Chi, and Yuxin Chen (2026). *Preconditioning Benefits of Spectral Orthogonalization in Muon*.

Marek, Martin, Sanae Lotfi, Aditya Somasundaram, Andrew Gordon Wilson, and Micah Goldblum (2025). "Small Batch Size Training for Language Models: When Vanilla SGD Works, and Why Gradient Accumulation is Wasteful". In: *Conference on Neural Information Processing Systems (NeurIPS)*.

Merity, Stephen, Caiming Xiong, James Bradbury, and Richard Socher (2017). "Pointer sentinel mixture models". In: *International Conference on Learning Representations (ICLR)*.

Michaud, Eric J, Ziming Liu, Uzay Girit, and Max Tegmark (2023). "The Quantization Model of Neural Scaling". In: *Conference on Neural Information Processing Systems (NeurIPS)*.

Mohanty, Saumyaranjan, Chimata Anudeep, and Konda Reddy Mopuri (2025). "Noise-free Loss Gradients: A Surprisingly Effective Baseline for Coreset Selection". In: *Transactions on Machine Learning Research (TMLR)*.

Nam, Yoonsoo, Nayara Fonseca, Seok Hyeong Lee, Chris Mingard, and Ard A. Louis (2024). "An exactly solvable model for emergence and scaling laws in the multitask sparse parity problem". In: *Conference on Neural Information Processing Systems (NeurIPS)*.

Nesterov, Yurii (2018). *Lectures on Convex Optimization*. Vol. 137. Springer Optimization and Its Applications. Springer.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research (JMLR)* 12, pp. 2825–2830.

Rosenfeld, Elan and Andrej Risteski (2024). "Outliers with Opposing Signals Have an Outsized Effect on Neural Network Optimization". In: *International Conference on Learning Representations (ICLR)*.

Sennrich, Rico, Barry Haddow, and Alexandra Birch (2016). "Neural machine translation of rare words with subword units". In: *Association for Computational Linguistics Conference (ACL)*.

Shah, Ishaan, Anthony M. Polloreno, Karl Stratos, et al. (2025). *Practical efficiency of Muon for pretraining*. arXiv preprint arXiv:2505.02222.

Srećković, Teodora, Jonas Geiping, and Antonio Orvieto (2025). "Is your batch size the problem? Revisiting the Adam-SGD gap in language modeling". In: *ICML workshop on High-dimensional Learning Dynamics (HILD)*.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin (2017). "Attention is all you need". In: *Conference on Neural Information Processing Systems (NeurIPS)*.

Vaswani, Sharan, Aaron Mishkin, Issam Hadj Laradji, Mark Schmidt, Gauthier Gidel, and Simon Lacoste-Julien (2019). "Painless Stochastic Gradient: Interpolation, Line-Search, and Convergence Rates". In: *Conference on Neural Information Processing Systems (NeurIPS)*.

Vyas, Nikhil, Depen Morwani, Rosie Zhao, Itai Shapira, David Brandfonbrener, Lucas Janson, and Sham M. Kakade (2025). "SOAP: Improving and Stabilizing Shampoo using Adam for Language Modeling". In: *International Conference on Learning Representations (ICLR)*.

Wang, Shuche, Fengzhuo Zhang, Jiaxiang Li, Cunxiao Du, Chao Du, Tianyu Pang, Zhuoran Yang, Mingyi Hong, and Vincent Y. F. Tan (2026). "Muon Outperforms Adam in Tail-End Associative Memory Learning". In: *International Conference on Learning Representations (ICLR)*.

Yin, Dong, Ashwin Pananjady, Max Lam, Dimitris Papailiopoulos, Kannan Ramchandran, and Peter Bartlett (2018). "Gradient Diversity: a Key Ingredient for Scalable Distributed Learning". In: *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Zhang, Tianyue H., Lucas Maes, Alan Milligan, Alexia Jolicoeur-Martineau, Ioannis Mitliagkas, Damien Scieur, Simon Lacoste-Julien, and Charles Guille-Escuret (2025a). "Understanding Adam requires better rotation dependent assumptions". In: *Conference on Neural Information Processing Systems (NeurIPS)*.

Zhang, Yushun, Congliang Chen, Ziniu Li, Tian Ding, Chenwei Wu, Diederik P. Kingma, Yinyu Ye, Zhi-Quan Luo, and Ruoyu Sun (2025b). "Adam-mini: Use fewer learning rates to gain more". In: *International Conference on Learning Representations (ICLR)*.

Zhao, Rosie, Depen Morwani, David Brandfonbrener, Nikhil Vyas, and Sham M. Kakade (2025). "Deconstructing what makes a good optimizer for autoregressive language models". In: *International Conference on Learning Representations (ICLR)*.

Zhuang, Zhenxun, Mingrui Liu, Ashok Cutkosky, and Francesco Orabona (2022). "Understanding AdamW through Proximal Methods and Scale-Freeness". In: *Transactions on Machine Learning Research (TMLR)*.

APPENDIX

A Proof for gradient descent under gradient cluster imbalance **15**

A.1 Proof of Theorem 1 for gradient descent 15

A.2 Proof of Theorem 2 for stochastic gradient descent 17

B Experimental setup **20**

B.1 Experimental details for Example 1 20

B.2 Experimental details for language models 20

B.3 Experimental details for vision 21

C Clustering Method **23**

C.1 Visualizing gradient cluster imbalance via gradient similarity. 23

C.2 Clustering with DBSCAN. 23

C.3 Clustering with HDBSCAN. 24

C.4 Reordering with seriation. 24

C.5 Sensitivity to clustering and reordering. 24

D Gaussian Check **26**

E Additional Structure **27**

E.1 Axis alignment in the output layer 27

E.2 Row and column orthogonality 27

F Evidence of gradient clusters and imbalance in language models **33**

F.1 Gradient cluster imbalance in a two-layer transformer 33

F.2 Gradient cluster imbalance in Pythia 1B 33

F.3 Semantics of the language model clusters 35

G Clustering and imbalance in vision **45**

G.1 Clustering across depth and training 45

G.2 Visual and semantic patterns within clusters 45

A Proof for gradient descent under gradient cluster imbalance

A.1 Proof of Theorem 1 for gradient descent

Let $\mathcal{L}$ be our objective function and $\mathcal{L}_i$ be the objective function on data from the i-th component,

$$\mathcal{L}(\mathbf{w}) := \sum_{i=1}^{m} \pi_i \mathcal{L}_i(\mathbf{w}), \qquad \mathbf{w} \in \mathbb{R}^d. \qquad \mathcal{L}_i(\mathbf{w}) := \mathop{\mathbb{E}}_{z \sim p_i}[\ell(\mathbf{w}; z)].$$

By Assumption 1, the objective function separates into m independent components.

Lemma 1 (Gradient orthogonality)**.** *Under Assumption 1, define the sample-gradient subspace of component i by*

$$\mathcal{S}_i := \operatorname{span}\left\{\nabla \ell(\mathbf{w}; z) : \mathbf{w} \in \mathbb{R}^d,\ z \in \mathcal{C}_i\right\}.$$

Let $\mathbf{P}_i \in \mathbb{R}^{d \times d}$ be the orthogonal projection onto $\mathcal{S}_i$:

$$\mathbf{P}_i \mathbf{w} \in \mathcal{S}_i, \qquad \mathbf{w} - \mathbf{P}_i \mathbf{w} \in \mathcal{S}_i^{\perp} \quad \textit{for every } \mathbf{w} \in \mathbb{R}^d.$$

The subspaces $\mathcal{S}_i$ are mutually orthogonal, so $\mathbf{P}_i \mathbf{P}_j = 0$ for $i \neq j$. Moreover, each sample loss from component i, and hence $\mathcal{L}_i$, depends only on $\mathbf{P}_i \mathbf{w}$:

$$\ell(\mathbf{w}; z) = \ell(\mathbf{P}_i \mathbf{w}; z) \quad (z \in \mathcal{C}_i), \qquad \mathcal{L}_i(\mathbf{w}) = \mathcal{L}_i(\mathbf{P}_i \mathbf{w}).$$

Proof. By Assumption 1, every sample gradient from component i is orthogonal to every sample gradient from component $j \neq i$. Their spans are therefore mutually orthogonal. For $z \in \mathcal{C}_i$, the gradient $\nabla \ell(\mathbf{w}; z)$ lies in $\mathcal{S}_i$ at every $\mathbf{w}$. Thus the loss is constant along directions in $\mathcal{S}_i^{\perp}$, giving $\ell(\mathbf{w}; z) = \ell(\mathbf{P}_i \mathbf{w}; z)$. Taking expectations over $z \sim p_i$ gives the corresponding identity for $\mathcal{L}_i$. □

As a result, gradient descent on $\mathcal{L}$ with step size η is equivalent to gradient descent on each component $\mathcal{L}_i$ with effective step size $\eta_i = \eta \pi_i$ and summing the resulting weights. The power-law assumption with power α for the frequencies gives the following $\pi_1, ..., \pi_m$,

$$\pi_i = \frac{i^{-\alpha}}{H_{m,\alpha}}, \qquad H_{m,\alpha} := \sum_{j=1}^{m} j^{-\alpha}, \qquad H_{0,\alpha} := 0$$

To analyze convergence, we start with the classical convergence rate of gradient descent on a convex, L-smooth function f with minimizer $\mathbf{w}^\star$ and minimum f^*, with a step size $0 < \eta \leq 2/L$ (Nesterov, 2018, Theorem 2.1.14);

$$f(\mathbf{w}_T) - f^\star \leq \frac{2\big(f(\mathbf{w}_0) - f^\star\big)\|\mathbf{w}_0 - \mathbf{w}^\star\|^2}{2\|\mathbf{w}_0 - \mathbf{w}^\star\|^2 + T\eta(2 - L\eta)\big(f(\mathbf{w}_0) - f^\star\big)}.$$

A slightly weaker but simpler bound is

$$\frac{f(\mathbf{w}_T) - f^\star}{f(\mathbf{w}_0) - f^\star} \leq \min\left\{1, \frac{2}{T\eta(2 - L\eta)} \frac{\|\mathbf{w}_0 - \mathbf{w}^\star\|^2}{f(\mathbf{w}_0) - f^\star}\right\}. \tag{1}$$

Theorem 1. *Under Condition 1 and Assumptions 1 and 2, gradient descent with step-size $\eta = 1/(\pi_1 L)$ guarantees a relative error of $\varepsilon \in (0, 1)$ within T iterations, where*

$$T = O\left(\begin{cases} m^{\alpha} & \textit{if } 0 \leq \alpha < 1 \\ m^{1 - c_1 \varepsilon} & \textit{if } \alpha = 1 \\ 1 & \textit{if } \alpha > 1 \end{cases}\right), \quad \textit{with} \quad c_1 := \frac{\Delta_{\min}}{\Delta_{\max}} \in (0, 1].$$

Here, big-O is asymptotic in terms of m and hides terms depending on $L, R, \Delta, \varepsilon$.

Proof of Theorem 1. By Assumption 1, subspaces are mutually orthogonal, and each $\mathcal{L}_i$ depends only on $\mathbf{P}_i \mathbf{w}$. Projecting the gradient descent update onto $\mathcal{S}_i$ gives

$$\mathbf{P}_i \mathbf{w}_{t+1} = \mathbf{P}_i \mathbf{w}_t - \eta \pi_i \nabla \mathcal{L}_i(\mathbf{P}_i \mathbf{w}_t).$$

Thus, component i follows gradient descent with effective step size $\eta_i = \eta\pi_i = \frac{\pi_i}{\pi_1 L} \le \frac{1}{L}$. Applying Equation (1) to $\mathcal{L}_i$ and bounding $\|\mathbf{P}_i(\mathbf{w}_0 - \mathbf{w}^\star)\| \le R$ gives

$$\mathcal{L}_i(\mathbf{w}_T) - \mathcal{L}_i^\star \le \frac{2\|\mathbf{P}_i(\mathbf{w}_0 - \mathbf{w}^\star)\|^2}{T\eta_i(2 - L\eta_i)} \le \frac{2R^2}{T\eta_i} = \frac{2\pi_1 LR^2}{\pi_i T},$$

Since $\eta_i \le 1/L$, the component objective values are nonincreasing and

$$\mathcal{L}_i(\mathbf{w}_T) - \mathcal{L}_i^\star \le \mathcal{L}_i(\mathbf{w}_0) - \mathcal{L}_i^\star.$$

Combining the two bounds gives

$$\mathcal{L}_i(\mathbf{w}_T) - \mathcal{L}_i^\star \le \min\left\{\mathcal{L}_i(\mathbf{w}_0) - \mathcal{L}_i^\star, \frac{2\pi_1 LR^2}{\pi_i T}\right\}.$$

Looking at the average error, we can use that bound to get

$$\begin{aligned}\frac{\mathcal{L}(\mathbf{w}_t) - \mathcal{L}^\star}{\mathcal{L}(\mathbf{w}_0) - \mathcal{L}^\star} &= \frac{\sum_i \pi_i(\mathcal{L}_i(\mathbf{w}_t) - \mathcal{L}_i^\star)}{\sum_i \pi_i(\mathcal{L}_i(\mathbf{w}_0) - \mathcal{L}_i^\star)}, \\ &\le \frac{1}{c_1}\sum_i \pi_i \frac{\mathcal{L}_i(\mathbf{w}_t) - \mathcal{L}_i^\star}{\mathcal{L}_i(\mathbf{w}_0) - \mathcal{L}_i^\star} \le \frac{1}{c_1}\sum_i \pi_i \min\left\{1, \frac{2\pi_1 LR^2}{\pi_i T\Delta_{\min}}\right\}.\end{aligned}$$

Substituting the power-law weights gives, for $T \ge 1$,

$$\frac{\mathcal{L}(\mathbf{w}_T) - \mathcal{L}^\star}{\mathcal{L}(\mathbf{w}_0) - \mathcal{L}^\star} \le \frac{1}{c_1}\sum_{i=1}^m \pi_i \min\left\{1, \frac{2\pi_1 LR^2}{\pi_i T\Delta_{\min}}\right\} = \frac{1}{c_1 H_{m,\alpha}}\sum_{i=1}^m \min\left\{i^{-\alpha}, \frac{2LR^2}{T\Delta_{\min}}\right\}.$$

For any integer $k \in \{0, \dots, m\}$, apply the convergence bound to the first k components and the initial-error bound to the remaining components. Then

$$\frac{\mathcal{L}(\mathbf{w}_T) - \mathcal{L}^\star}{\mathcal{L}(\mathbf{w}_0) - \mathcal{L}^\star} \le \frac{2LR^2 k/(T\Delta_{\min}) + H_{m,\alpha} - H_{k,\alpha}}{c_1 H_{m,\alpha}}.$$

We now choose k and T to make this bound at most ε, for fixed $L, R, \Delta_{\min}, \Delta_{\max}, \alpha, \varepsilon$ as m grows.

Case $0 \le \alpha < 1$**.** Choose $k = m$, so the tail contribution vanishes. Since $H_{m,\alpha} \ge m^{1-\alpha}$, we obtain

$$\frac{\mathcal{L}(\mathbf{w}_T) - \mathcal{L}^\star}{\mathcal{L}(\mathbf{w}_0) - \mathcal{L}^\star} \le \frac{2LR^2 m}{c_1 H_{m,\alpha} T\Delta_{\min}} \le \frac{2LR^2 m^\alpha}{c_1 T\Delta_{\min}}, \quad \text{giving} \quad T_{\mathrm{GD}}(\varepsilon) \le \left\lceil \frac{2LR^2}{c_1\Delta_{\min}\varepsilon} m^\alpha \right\rceil = O(m^\alpha).$$

Case $\alpha = 1$**.** For $1 \le k \le m$, the integral bound gives

$$H_{m,1} - H_{k,1} = \sum_{i=k+1}^m \frac{1}{i} \le \int_k^m \frac{dx}{x} = \log(m/k).$$

Choose $k = \lceil e\, m^{1-c_1\varepsilon} \rceil$, and $T = \lceil 2LR^2 k/\Delta_{\min} \rceil$. The factor e leaves room for the error of the first k components. For sufficiently large m, we have $k \le m$ and

$$\frac{2LR^2 k}{T\Delta_{\min}} \le 1, \qquad \log(m/k) \le c_1\varepsilon \log m - 1.$$

Since $H_{m,1} \ge \log m$, it follows that

$$\frac{\mathcal{L}(\mathbf{w}_T) - \mathcal{L}^\star}{\mathcal{L}(\mathbf{w}_0) - \mathcal{L}^\star} \le \frac{1 + \log(m/k)}{c_1 H_{m,1}} \le \frac{c_1\varepsilon \log m}{c_1 H_{m,1}} \le \varepsilon, \qquad \text{and} \qquad T_{\mathrm{GD}}(\varepsilon) = O(m^{1-c_1\varepsilon}).$$

Case $\alpha > 1$**.** Choose a cutoff $K \ge 1$ independent of m such that

$$\frac{K^{1-\alpha}}{\alpha - 1} \le \frac{c_1\varepsilon}{2},$$

and set $k = \min\{K, m\}$. If $m > K$, the tail satisfies

$$H_{m,\alpha} - H_{K,\alpha} \le \int_K^\infty x^{-\alpha}\, dx = \frac{K^{1-\alpha}}{\alpha - 1} \le \frac{c_1\varepsilon}{2}.$$

If $m \le K$, the tail is zero. Choosing $T = \left\lceil \frac{4LR^2K}{c_1\Delta_{\min}\varepsilon} \right\rceil$ gives

$$\frac{2LR^2k}{T\Delta_{\min}} \le \frac{c_1\varepsilon}{2}.$$

Since $H_{m,\alpha} \ge 1$, we obtain

$$\frac{\mathcal{L}(\mathbf{w}_T) - \mathcal{L}^\star}{\mathcal{L}(\mathbf{w}_0) - \mathcal{L}^\star} \le \frac{c_1\varepsilon/2 + c_1\varepsilon/2}{c_1 H_{m,\alpha}} \le \varepsilon.$$

Both K and T are independent of m, so $T_{\mathrm{GD}}(\varepsilon) = O(1)$. □

A.2 Proof of Theorem 2 for stochastic gradient descent

Assume Conditions 1 and 2 and assumption 2, and Assumption 1. At each iteration, independently sample a component I_t with $\mathbb{P}(I_t = i) = \pi_i$, then draw $z_t \sim p_i$. SGD updates

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \nabla \ell(\mathbf{w}_t; z_t).$$

We use the constant step size $\eta = 1/L$ and study the averaged iterate

$$\bar{\mathbf{w}}_T := \frac{1}{T}\sum_{t=0}^{T-1} \mathbf{w}_t.$$

Let S_i and P_i denote the component subspaces and orthogonal projections of component i from Lemma 1. By Assumption 2, the initial component distances are bounded by R, and the initial component errors lie in $[\Delta_{\min}, \Delta_{\max}]$.

To exploit the component frequencies, we establish a component bound that accounts for the random times at which each component is sampled.

Lemma 2 (Component error under interpolating SGD)**.** *Under Conditions 1 and 2, and Assumptions 1 and 2, SGD with step size $\eta = 1/L$ satisfies, for every component i and integer $T \ge 1$,*

$$\mathbb{E}[\mathcal{L}_i(\bar{\mathbf{w}}_T) - \mathcal{L}_i^\star] \le \min\left\{ \frac{LR^2}{2}, \frac{LR^2}{2\pi_i T} \right\}.$$

Proof. By Assumption 1, these subspaces are mutually orthogonal, and each sample loss from component i depends only on $\mathbf{P}_i\mathbf{w}$. Let $\mathbf{w}_i^\star$ be a minimizer of $\mathcal{L}_i$ satisfying Condition 2, and write

$$\mathbf{u}_{i,t} := \mathbf{P}_i\mathbf{w}_t, \qquad \mathbf{u}_i^\star := \mathbf{P}_i\mathbf{w}_i^\star.$$

By interpolation, almost every sample loss attains its minimum at the common minimizer from Condition 3. Since $\mathbf{w}_i^\star$ also minimizes their expectation, it minimizes almost every sample loss in component i. Moreover,

$$\|\mathbf{u}_{i,0} - \mathbf{u}_i^\star\| \le \|\mathbf{w}_0 - \mathbf{w}_i^\star\| \le R.$$

At iteration t, let I_t denote the sampled component and $z_t \sim p_{I_t}$ the sample drawn from that component. The projected SGD update is

$$\mathbf{u}_{i,t+1} = \mathbf{u}_{i,t} - \eta \mathbf{1}_{\{I_t = i\}} \nabla \ell(\mathbf{u}_{i,t}; z_t).$$

When $I_t = i$, convexity, L-smoothness, and interpolation give

$$\langle \nabla \ell(\mathbf{u}_{i,t}; z_t), \mathbf{u}_{i,t} - \mathbf{u}_i^\star \rangle \ge \ell(\mathbf{u}_{i,t}; z_t) - \ell(\mathbf{u}_i^\star; z_t) + \frac{\|\nabla \ell(\mathbf{u}_{i,t}; z_t)\|^2}{2L}.$$

Expanding the squared distance and using $\eta = 1/L$ therefore yields

$$\|\mathbf{u}_{i,t+1} - \mathbf{u}_i^\star\|^2 \le \|\mathbf{u}_{i,t} - \mathbf{u}_i^\star\|^2 - 2\eta\big(\ell(\mathbf{u}_{i,t}; z_t) - \ell(\mathbf{u}_i^\star; z_t)\big).$$

Let $\mathcal{F}_t$ denote the sampling history before iteration t. Applying this inequality when component i is sampled, and noting that its iterate is unchanged otherwise, gives

$$\mathbb{E}\Big[\|\mathbf{u}_{i,t+1} - \mathbf{u}_i^\star\|^2 \mid \mathcal{F}_t\Big] \le \|\mathbf{u}_{i,t} - \mathbf{u}_i^\star\|^2 - 2\eta\pi_i\big(\mathcal{L}_i(\mathbf{u}_{i,t}) - \mathcal{L}_i^\star\big).$$

Taking expectations and summing over $t = 0, \dots, T-1$,

$$2\eta\pi_i \sum_{t=0}^{T-1} \mathbb{E}[\mathcal{L}_i(\mathbf{u}_{i,t}) - \mathcal{L}_i^\star] \le \|\mathbf{u}_{i,0} - \mathbf{u}_i^\star\|^2 \le R^2.$$

Convexity therefore yields

$$\mathbb{E}[\mathcal{L}_i(\bar{\mathbf{w}}_T) - \mathcal{L}_i^\star] \le \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}[\mathcal{L}_i(\mathbf{u}_{i,t}) - \mathcal{L}_i^\star] \le \frac{LR^2}{2\pi_i T}.$$

Finally, interpolation makes each sample-loss gap nonnegative, so the distance inequality also implies

$$\|\mathbf{u}_{i,t} - \mathbf{u}_i^\star\| \le R \qquad \text{almost surely for every } t.$$

The averaged component lies in the same ball. By smoothness of $\mathcal{L}_i$,

$$\mathcal{L}_i(\bar{\mathbf{w}}_T) - \mathcal{L}_i^\star \le \frac{L}{2}\|\mathbf{P}_i\bar{\mathbf{w}}_T - \mathbf{u}_i^\star\|^2 \le \frac{LR^2}{2}.$$

Combining the two bounds proves the claim. □

Theorem 2. *Under Assumptions 1 and 2 and Conditions 1 and 2, SGD with step size $1/L$ reaches expected relative population error ε at its averaged iterate within*

$$T = O\left(\begin{cases} m & \text{if } 0 \le \alpha < 1 \\ m^{1-c_2\varepsilon}\log m & \text{if } \alpha = 1 \\ 1 & \text{if } \alpha > 1 \end{cases}\right), \qquad \text{with} \qquad c_2 := \frac{2\Delta_{\min}}{LR^2} \in (0, 1].$$

Here, big-O is asymptotic in terms of m and hides terms depending on $L, R, \Delta, \varepsilon$.

Proof of Theorem 2. Define the expected relative population error and its iteration requirement by

$$\mathcal{E}_T^{\mathrm{SGD}} := \frac{\mathbb{E}[\mathcal{L}(\bar{\mathbf{w}}_T) - \mathcal{L}^\star]}{\mathcal{L}(\mathbf{w}_0) - \mathcal{L}^\star}, \qquad T_{\mathrm{SGD}}(\varepsilon) := \min\{T \in \mathbb{Z}_{\ge 1} : \mathcal{E}_T^{\mathrm{SGD}} \le \varepsilon\}.$$

By interpolation, the component minima are attained simultaneously. Condition 2 therefore gives

$$\mathcal{L}(\mathbf{w}_0) - \mathcal{L}^\star = \sum_{i=1}^m \pi_i\big(\mathcal{L}_i(\mathbf{w}_0) - \mathcal{L}_i^\star\big) \ge \Delta_{\min}.$$

Summing Lemma 2 with weights π_i and dividing by the initial population error yields

$$\mathcal{E}_T^{\mathrm{SGD}} \le \frac{LR^2}{2\Delta_{\min}} \sum_{i=1}^m \min\left\{\pi_i, \frac{1}{T}\right\} = \frac{1}{c_2}\sum_{i=1}^m \min\left\{\pi_i, \frac{1}{T}\right\}.$$

Smoothness and Assumption 2 imply $\Delta_{\min} \le LR^2/2$, so $0 < c_2 \le 1$. For any integer $k \in \{0, \dots, m\}$, apply the convergence bound to the first k components and the distance-based cap to the remaining components. Then

$$\mathcal{E}_T^{\mathrm{SGD}} \le \frac{1}{c_2}\left(\frac{k}{T} + \frac{H_{m,\alpha} - H_{k,\alpha}}{H_{m,\alpha}}\right). \tag{2}$$

We now choose k and T in each regime so that this bound is at most ε.

Case $0 \le \alpha < 1$**.** Choose $k = m$, so the tail contribution vanishes. Then

$$\mathcal{E}_T^{\mathrm{SGD}} \le \frac{m}{c_2 T} \qquad \text{and} \qquad T_{\mathrm{SGD}}(\varepsilon) \le \left\lceil \frac{m}{c_2\varepsilon} \right\rceil = O(m).$$

Case $\alpha = 1$**.** For $1 \le k \le m$,

$$H_{m,1} - H_{k,1} \le \int_k^m \frac{dx}{x} = \log(m/k).$$

Choosing $k = \lceil e\, m^{1-c_2\varepsilon} \rceil$ and $T = \lceil kH_{m,1} \rceil$, for sufficiently large m, we have $k \le m$, and

$$\frac{k}{T} \le \frac{1}{H_{m,1}}, \qquad \log(m/k) \le c_2\varepsilon \log m - 1.$$

Substituting into Equation (2) gives

$$\mathcal{E}_T^{\mathrm{SGD}} \le \frac{1 + \log(m/k)}{c_2 H_{m,1}} \le \varepsilon \frac{\log m}{H_{m,1}} \le \varepsilon.$$

Since $H_{m,1} = O(\log m)$, we have $T_{\mathrm{SGD}}(\varepsilon) \le T = O(m^{1-c_2\varepsilon} \log m)$.

Case $\alpha > 1$. Choose a cutoff $K \ge 1$ independent of m such that

$$\frac{K^{1-\alpha}}{\alpha - 1} \le \frac{c_2\varepsilon}{2},$$

and set $k = \min\{K, m\}$. If $m > K$, then

$$H_{m,\alpha} - H_{K,\alpha} \le \int_K^\infty x^{-\alpha}\, dx = \frac{K^{1-\alpha}}{\alpha - 1} \le \frac{c_2\varepsilon}{2}.$$

If $m \le K$, the tail is zero. Choose $T = \left\lceil \frac{2K}{c_2\varepsilon} \right\rceil$. Since $k \le K$ and $H_{m,\alpha} \ge 1$, we obtain

$$\mathcal{E}_T^{\mathrm{SGD}} \le \frac{1}{c_2} \left(\frac{c_2\varepsilon}{2} + \frac{c_2\varepsilon}{2} \right) = \varepsilon.$$

Both K and T are independent of m, so $T_{\mathrm{SGD}}(\varepsilon) = O(1)$. □

B EXPERIMENTAL SETUP

This section details the setup for the experiments studied in the paper: the synthetic minimal example of Example 1 (Section B.1), language models, where we train a two-layer transformer, and also analyze released checkpoints of Pythia-1B (Section B.2), and vision, where we train a ResNet-20 on CIFAR-100 (Section B.3).

B.1 EXPERIMENTAL DETAILS FOR EXAMPLE 1

Optimizers and learning rates. We compare SGD, Adam with $(\beta_1, \beta_2) = (0, 0.999)$ and $\epsilon = 10^{-8}$, and Muon. All optimizers use constant learning rates and no weight decay; SGD and Muon use zero momentum. With the exact gradient, SGD reduces to gradient descent, which we label GD. For each problem, batch size, and optimizer, we run every learning rate in a candidate grid and report the stable run with the lowest final training loss. Learning rates are selected from problem- and batch-specific candidate sets, including logarithmically spaced grids and local refinements.

Effect of the Newton–Schulz approximation. We use PyTorch's `torch.optim.Muon` with default settings ($k = 5$ Newton–Schulz steps). In Example 1, the gradient is already in SVD form, $\nabla L(W) = \sum_i \pi_i r_i \, u_i v_i^\top$ with $r_i = u_i^\top W v_i - 1$, so exact spectral descent would update all components at the same rate regardless of π_i. A possible cause of Muon's slower tail convergence is the finite Newton–Schulz approximation. It normalizes the gradient by its Frobenius norm and applies k iterations of an odd polynomial $p(x) = ax + bx^3 + cx^5$. Since $p(x) \approx ax$ near zero, small singular values are amplified by at most $a^k \approx 490$ and may remain well below 1. Under power-law frequencies, the Frobenius norm is dominated by frequent components, so rare components may fall in this regime and receive updates that still scale with π_i. As frequent components converge, the Frobenius norm shrinks and the rare components' updates approach full size, which is consistent with the delayed but rapid tail convergence in Figure 3.

Problems and batch sizes. Unless stated otherwise, class probabilities are $\pi_i = (1/i)/H_m$, the matrix dimension is $m = 10000$, and we train for $T = 200$ iterations from $W_0 = 0$ with a fixed random seed. The rotated diagonal problem uses fixed orthogonal matrices U, V obtained from QR decompositions of Gaussian matrices. The diagonal and rotated-diagonal comparisons use the exact full gradient. Finite minibatches are sampled with replacement. We log the full training loss and group losses at every iteration. Classes, ordered by decreasing sampling probability, are partitioned into five contiguous groups of approximately equal probability mass. Within each group, losses are averaged with sampling-probability weights.

GD versus SGD across frequency exponents. For Figure 2, we run GD and SGD on Example 1 with $\pi_k \propto k^{-\alpha}$ and $W_0 = 0$, so that $L(W_0) = \frac{1}{2}$. Since the matrices $u_k v_k^\top$ are orthonormal and GD and SGD are rotation-equivariant, the problem is equivalent to the diagonal quadratic $L(w) = \frac{1}{2}\sum_{k=1}^m \pi_k (w_k - 1)^2$ in the coordinates $w_k = u_k^\top W v_k$, which we use for efficiency. We vary $\alpha \in [0, 2]$ in steps of 0.05 for $m \in \{10^3, 10^4, 10^5\}$. GD uses the exact gradient with a constant step-size, tuned separately for each horizon within the stability interval $(0, 2/\pi_1)$. SGD samples one component per step with replacement and uses $\eta = 1$, which solves the sampled component exactly and minimizes the expected loss at every horizon. Both losses are computed in closed form, $L_t = \frac{1}{2}\sum_k \pi_k (1 - \eta\pi_k)^{2t}$ for GD and $\mathbb{E}[L_t] = \frac{1}{2}\sum_k \pi_k (1 - \pi_k)^t$ for SGD, so no Monte Carlo estimates are involved. We report the loss after $1{,}000$ updates and the smallest number of updates at which the relative loss satisfies $L_t / L(W_0) \le 0.2$; for SGD, this threshold applies to the expected loss rather than to individual trajectories. Note that equal numbers of updates do not correspond to equal computational cost, since each GD update evaluates all m components.

B.2 EXPERIMENTAL DETAILS FOR LANGUAGE MODELS

Two-layer transformer: model, data, and training. We use a small transformer model (Vaswani et al., 2017) with depth 2 (two transformer blocks), sequence length of 32, embedding width of

256 (and 1024 for the hidden dimension of the MLP), 2 heads and dropout of 0.1. We use the implementation from the word language model example from PyTorch.[2]

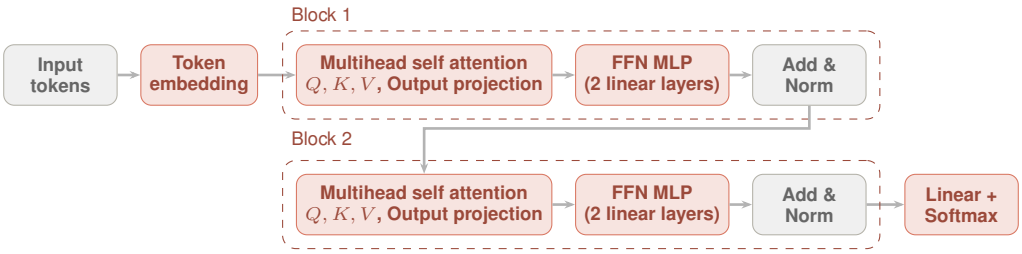

Figure 6: Two modules transformer model architecture.

We use the WikiText-103 dataset (Merity et al., 2017), tokenized using Byte-Pair Encoding (Sennrich et al., 2016) with a vocabulary size of $2^{14} = 16\,384$.

Training uses Adam with default hyperparameters except for the step-size, which is set to $10^{-2.5}$, selected by grid-search over half-powers of 10 and a batch size (number of sequences) of 512. We do not use a learning rate schedule.

Pythia-1B: model, data, and checkpoints. We study a larger transformer language model from the Pythia suite (Biderman et al., 2023): Pythia-1B, with 16 transformer blocks, model width 2,048, MLP width 8,192, and 8 attention heads. The model have a maximum context length of 2,048 and use the GPT-NeoX byte-pair encoding tokenizer with a vocabulary of 50,304 tokens. We use the checkpoints released by EleutherAI without additional training or fine-tuning.

The model was pretrained on the Pile (Gao et al., 2020) for 143,000 optimizer steps, or approximately 300 billion tokens, with a global batch size of 2,097,152 tokens. Pretraining used Adam with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$, weight decay 0.1, and gradient clipping at norm 1.0. The peak learning rates is 2.5×10^{-4}. The learning rate was warmed up over the first 1% of training, then cosine-decayed to one tenth of its peak value. The model is pretrained in BF16 and does not use dropout. We analyze checkpoints at initialization and at steps $1{,}000$, $18{,}000$, $36{,}000$, $54{,}000$, and $72{,}000$. Step $72{,}000$ is approximately the midpoint of training.

Computing per-sample gradients In a language model, each token position defines a separate next-token prediction problem. For example, the sequence $abcd$ gives the prediction samples (a, b), (ab, c), and (abc, d). For all checkpoints, we use the same fixed set of 128 sequences of length 32 from the Pile validation split. We select 16 alternating prediction positions per sequence, $0, 2, \ldots, 30$, yielding $128 \times 16 = 2{,}048$ token-level samples. For each selected position, we compute the unreduced next-token cross-entropy loss and differentiate it independently with respect to the selected parameters. We load the checkpoints in FP32 and evaluate them without key-value caching.

We compute gradients for the input embedding, the output head, and, in each transformer block, the attention Q, K, V, and output projections and the MLP up- and down-projections. To reduce storage, we flatten each attention and MLP gradient into a vector and compress it to 2,048 dimensions using CountSketch. For each matrix, we fix the sketch and reuse it across samples and checkpoints, so that cosine similarities are approximated consistently. For the input embedding and output head, we instead use compact representations that preserve the gradients exactly. For the input embedding, we store only the nonzero rows, which correspond to the tokens in the context. Each output-head gradient has the form $(p - e_y)h^\top$, where p is the vector of predicted probabilities, e_y is the one-hot encoding of the target, and h is the final hidden state; we store the two vectors $p - e_y$ and h instead of the full matrix.

B.3 Experimental details for vision

Model and training. We train ResNet-20 (He et al., 2016) on CIFAR-100 (Krizhevsky, 2009) from scratch on the 50,000 training images of CIFAR-100, comprising 100 classes of 32×32 RGB images.

[2] https://github.com/pytorch/examples/tree/acc295dc7b90714f1bf47f06004fc19a7fe235c4/word_language_model

The network has three residual stages with 16, 32, and 64 channels, respectively, and three residual blocks per stage. Each block contains two 3×3 convolutions with batch normalization and ReLU activations; stage transitions use learned projection shortcuts. Global average pooling is followed by a 100-class linear classifier.

We train for 50 epochs using cross-entropy loss and SGD with batch size 128, momentum 0.9, weight decay 5×10^{-4}, and an initial learning rate of 0.1, cosine-annealed to zero. Training images undergo random cropping with four-pixel padding, random horizontal flipping, and channel-wise normalization. The model reaches approximately 67% test accuracy. We use this training budget to study how gradient structure emerges and evolves during early and middle training, rather than to maximize final accuracy. We analyze initialization and checkpoints after epochs 5, 8, 15, 30, and 50.

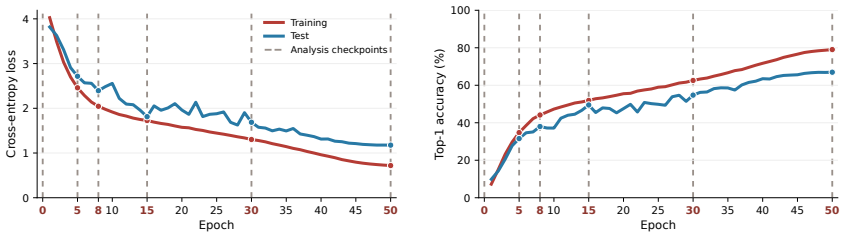

Figure 7: Training and test loss and accuracy for ResNet-20 on CIFAR-100. Dashed lines mark checkpoints used for gradient analysis. The model reaches about 67% test accuracy after 50 epochs.

Per-sample gradients. We select a fixed, class-balanced subset of 2,000 test images, sampling 20 images per class without replacement and shuffling the subset. The same images are used across layers and checkpoints. During gradient extraction, the model is in evaluation mode, with fixed batch-normalization statistics and no random augmentation. For each image, we compute the gradient of its cross-entropy loss with respect to the selected layer's weights. We flatten these gradients in their original parameter coordinates and measure pairwise absolute cosine similarity.

C CLUSTERING METHOD

Throughout, we compare per-sample gradients using the absolute cosine similarity $S_{ij} = |\cos(g_i, g_j)|$, so aligned and opposite gradients are treated as similar. Clusters are fitted independently at each checkpoint, so they are not tracked over time, and blocks at the same position need not contain the same samples. Unless stated otherwise, figures use DBSCAN and display the first 500 reordered samples, while reported cluster sizes refer to the full sample set. Clustering hyperparameters and the displayed subset can affect how prominent blocks appear, so we treat comparisons across layers and checkpoints as descriptive. We then describe two alternatives, HDBSCAN and seriation, and show that when cluster structure exists, the main patterns are robust to this choice. We first illustrate what the similarity matrix looks like under exact gradient cluster imbalance.

C.1 VISUALIZING GRADIENT CLUSTER IMBALANCE VIA GRADIENT SIMILARITY.

We use a synthetic example to illustrate how gradient cluster imbalance appears in gradient-similarity plots. We draw 500 samples from 150 categories with probabilities $\pi_i \propto i^{-\alpha}$. Each category is assigned a unit-norm gradient orthogonal to those of all other categories, and each sample inherits its category's gradient. Pairwise cosine similarity is therefore 1 within categories and 0 across categories. The top row of Figure 8 shows category probabilities for the first 25 category ranks, while the bottom row shows the corresponding gradient-similarity matrices with samples grouped by category.

As α increases from 0 to 2.5, the distribution becomes increasingly concentrated: the many small diagonal blocks under uniform sampling give way to a few large blocks representing frequent categories and many smaller blocks representing rare categories. The middle panel ($\alpha = 1$) illustrates Zipf-like frequencies, with a prominent leading block followed by progressively smaller blocks. In real models, we reorder samples using clustering to reveal analogous structure. A pattern resembling the middle panel would be qualitatively consistent with Zipf-like gradient cluster imbalance, although visual similarity alone does not establish a power-law distribution.

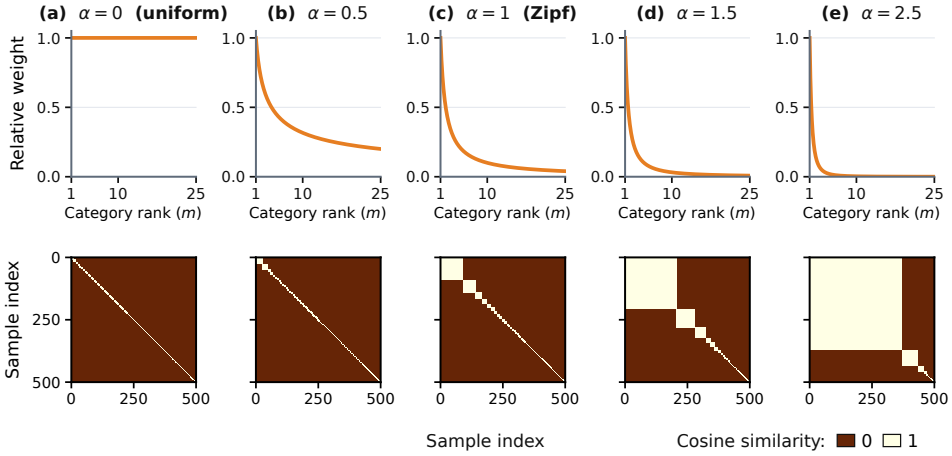

Figure 8: Effect of frequency imbalance on sample gradient similarity. Top: category probabilities $\pi_i \propto i^{-\alpha}$, ranked in decreasing order. Bottom: sample gradient cosine-similarity matrices, with samples grouped by category in the same order. Larger α concentrates probability in the top categories, producing more imbalanced diagonal blocks.

C.2 CLUSTERING WITH DBSCAN.

To find the permutation matrix to visualize the block-diagonal structure, we use the Scikit-learn (Pedregosa et al., 2011) implementation of DBSCAN (Ester et al., 1996) using (absolute) cosine similarities as the metric. We form a permutation matrix $\mathbf{P}$ by sorting the clusters in decreasing order of a combination of size and intensity of within-cluster similarity. For a cluster $\mathcal{S} \subseteq [n]$ and a

similarity matrix $\mathbf{S} \in [0,1]^{n \times n}$, the score of a cluster is given by

$$\text{Score}(\mathcal{S}) = \frac{1}{|\mathcal{S}|} \sum_{i,j \in \mathcal{S}} |\mathbf{S}[i,j]|.$$

We tune the ϵ parameter controlling the neighborhood size automatically by a heuristic, minimizing a score over the fixed grid of $\epsilon = [1/1000, 1/5, 2/5, 3/5, 4/5, 1]$. Given a permutation matrix $\mathbf{P}$ induced by the clusters $\mathcal{S}_1, \ldots, \mathcal{S}_k$ obtained by DBSCAN with a given ϵ, we score the value of ϵ using

$$\text{Score}(\epsilon) = \frac{1}{n-1} \sum_{i=1}^{n-1} 1 - (\mathbf{P}\mathbf{S}\mathbf{P}^\top)[i, i+1] + \frac{|\mathcal{S}_1|^2}{n^2}.$$

The first term is the normalized path length as a measure of the quality of the seriation and the second is the percentage of the area occupied by the largest cluster to avoid solutions where the first cluster contains too many samples. Beyond those intuitions, we do not have a strong motivation for the heuristic other than that we found the results to be consistent across layers.

C.3 Clustering with HDBSCAN.

We use the sklearn (Pedregosa et al., 2011) implementation of HDBSCAN (Campello et al., 2013) with the precomputed absolute cosine similarity of gradient pairs. HDBSCAN builds a hierarchy of density-based clusters and identifies groups that persist across density levels, rather than using a single neighborhood radius. For each checkpoint and parameter matrix, we sweep `min_cluster_size`, the minimum number of samples in a cluster, over $\{5, 10, 20, 41\}$, and `min_samples`, the number of neighbors used to estimate local density, over $\{5, 10, 20\}$.

We use the excess-of-mass (`eom`) cluster-selection method. We choose the parameter setting using a score that combines the fraction of samples assigned to clusters, their silhouette coefficient, the mean HDBSCAN membership probability, and a soft penalty for very small or large numbers of clusters.

To form the permutation matrix $\mathbf{P}$, we order clusters from largest to smallest, breaking ties by decreasing cluster label. Each assigned sample also receives a membership probability in $[0, 1]$, the density level at which it leaves its cluster relative to the cluster's densest point; core samples have probability 1 and peripheral samples lower values. Within each cluster, we order samples by decreasing membership probability. Unassigned samples (label-1) are placed last, in their original sample-index order.

C.4 Reordering with seriation.

We use hierarchical seriation (Bar-Joseph et al., 2001) to place samples with similar gradients near one another, revealing block structure without explicit cluster assignments. We use average-linkage hierarchical clustering followed by optimal leaf ordering, both implemented in `scipy.cluster.hierarchy`. For per-sample gradients $\mathbf{g}_i$, we compute the absolute cosine similarities. Before constructing the tree, we randomly permute the samples using a fixed seed of 1957 plus the parameter-matrix index. This reduces systematic dependence on the original sample order when ties occur. Using the precomputed distances, we construct the tree with `linkage(method="average")`, then apply `optimal_leaf_ordering` to minimize the sum of distances between adjacent samples while preserving the tree structure. The resulting order defines the permutation matrix $\mathbf{P}$, which we apply to both axes:

$$\mathbf{S}_{\text{ordered}} = \mathbf{P}\mathbf{S}\mathbf{P}^\top.$$

Seriation orders all samples without assigning cluster labels or marking any samples as noise.

C.5 Sensitivity to clustering and reordering.

Figure 9 compares DBSCAN, HDBSCAN, and seriation on selected Pythia-1B layers and checkpoints. All three reveal clear blocks where clustering is strong, while weakly structured examples remain less distinct across methods. The main patterns therefore do not depend on a particular reordering method,

although block order and apparent boundaries vary. DBSCAN and HDBSCAN also provide explicit cluster assignments, making it convenient to group samples for visualization and further analysis.

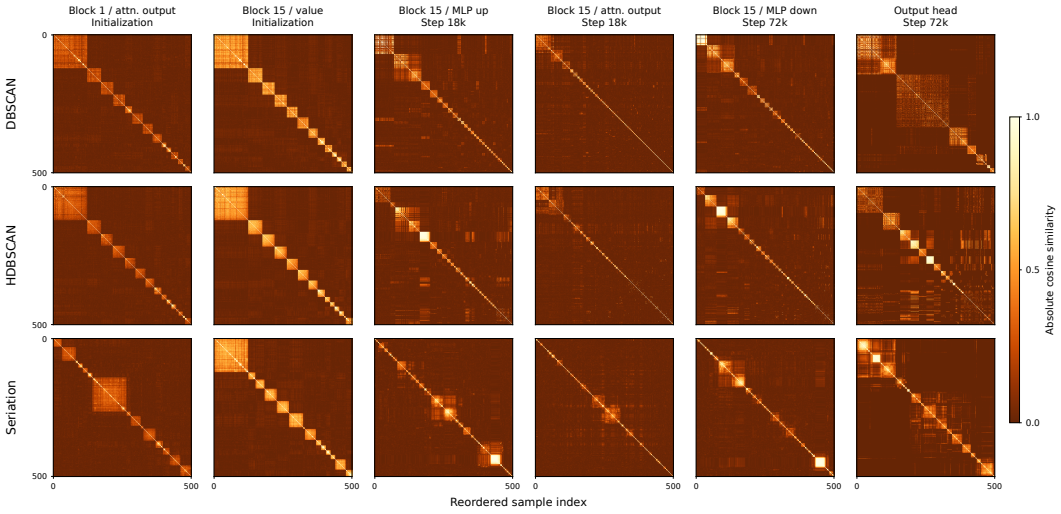

Figure 9: **Gradient grouping is visible across different clustering and reordering methods.** Columns show selected Pythia-1B layers and checkpoints; rows show DBSCAN, HDBSCAN, and seriation. Colors indicate absolute cosine similarity. Strong block structure remains visible across methods, while block order and apparent boundaries vary.

D GAUSSIAN CHECK

Clustering and reordering such as DBSCAN always returns a partition, and sorting samples by cluster concentrates large entries near the diagonal. To check that the blocks we observe reflect the gradients actually exist, we compare against a null model passed through the same pipeline.

We use a Gaussian with the same mean and covariance as the per-sample gradients. This null preserves all first- and second-order structure, including any imbalance in the covariance spectrum. Gradient cluster imbalance implies such an imbalance: with orthogonal clusters, the second moment is block-diagonal, with blocks weighted by the cluster frequencies π_k. The converse does not hold. A Gaussian with an imbalanced spectrum has the same dominant directions, but its samples spread continuously across them rather than concentrating in distinct groups. If the blocks were explained by the mean and covariance alone, they should also appear in the Gaussian samples.

For each layer and checkpoint, we compute the empirical mean μ and covariance Σ of the n per-sample gradients in the representation used for clustering, and draw n samples from $\mathcal{N}(\mu, \Sigma)$. Since n is smaller than the dimension, Σ is rank-deficient, and we sample $\tilde{g} = \mu + \frac{1}{\sqrt{n-1}} \sum_{i=1}^{n} z_i (g_i - \mu)$ with $z \sim \mathcal{N}(0, I_n)$, which matches the empirical mean and covariance exactly. We then cluster, reorder, and display the real and Gaussian samples independently using DBSCAN.

Figures 10 and 11 show clear blocks in the real gradients for the layers shown in both language and vision models, but no comparable blocks in their Gaussian controls. The observed grouping is therefore not an artifact of the clustering procedure, and cannot be explained by the mean and covariance of the gradients alone. This control does not rule out every unimodal alternative, but it rules out explanations based only on the dominant directions of the gradient covariance.

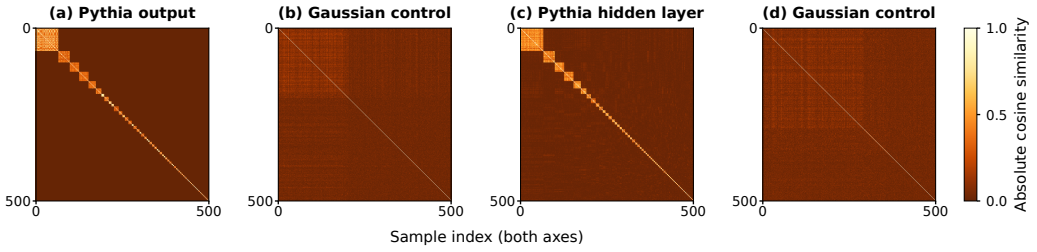

Figure 10: **Matched Gaussian samples do not reproduce gradient clusters.** Absolute cosine similarity for Pythia-1B output-layer (a) and Block 7 attention-output (c) gradients, alongside their matched Gaussian controls (b, d) with their respective mean and covariance. Each matrix is independently reordered using DBSCAN; 500 samples are displayed.

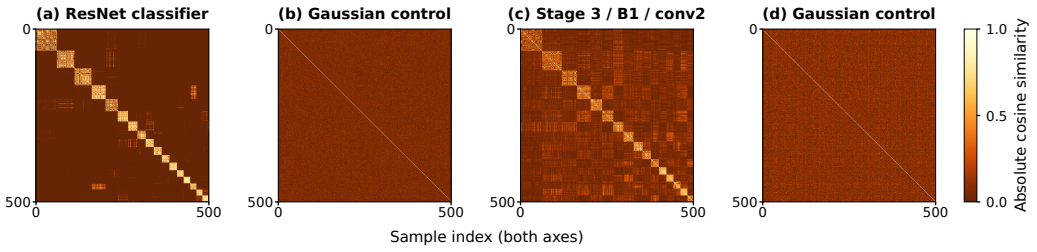

Figure 11: ResNet-20 classifier and Stage-3, block-1, conv2 gradient similarities at epoch 30, compared with Gaussian controls matching their mean and covariance. Each matrix is independently reordered using DBSCAN; 500 samples are displayed.

E ADDITIONAL STRUCTURE

The benefits of sign and spectral methods rely on structural assumptions beyond gradient cluster imbalance: sign descent removes the frequency weights when clusters are axis-aligned (Assumption 3), and spectral descent when they are spectrally aligned (Assumption 4). Both assumptions are stated for all parameter values and the whole network, so they are too strong to hold exactly in practice. We therefore test whether they hold approximately for individual layers at selected checkpoints. For axis alignment, we examine the output layer, where rows of the weight matrix correspond directly to output tokens or classes and where Adam is typically retained. For row and column orthogonality, we examine hidden layers, where Muon is typically applied.

E.1 AXIS ALIGNMENT IN THE OUTPUT LAYER

To examine Assumption 3, we measure how each sample's gradient is distributed across output-weight rows. For a linear output layer with cross-entropy loss, the gradient for sample i is $\mathbf{G}_i = (\mathbf{p}_i - \mathbf{e}_{y_i})\mathbf{h}_i^\top$, where $\mathbf{p}_i$ is the predicted probability vector, $\mathbf{e}_{y_i}$ is the one-hot target, and $\mathbf{h}_i$ is the hidden representation. Each plotted entry is $N_{ia} = \|\mathbf{G}_i[a,:]\|_2$, the gradient norm for sample i on the weight row corresponding to output token or class a. Samples are grouped by target along the y-axis, and output tokens or classes appear along the x-axis.

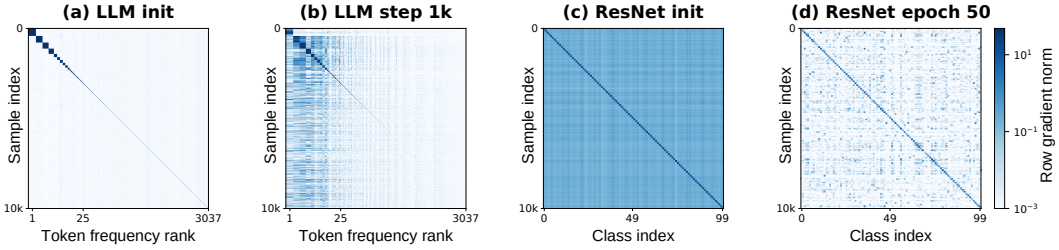

Figure 12: **Output-layer gradients often concentrate on the target's parameter row.** Dark diagonal regions indicate large gradients on target rows; pale off-diagonal regions indicate small gradients on other rows, supporting approximate coordinate separation (axis alignment) of the last layer. Each entry shows a sample's gradient norm for one output-weight row, on a shared logarithmic scale clipped below at 10^{-3}. Samples are grouped by target output, with column widths proportional to target frequency. Targets are ordered by decreasing frequency in language and by class ID in vision.

In Figure 12, pale off-diagonal regions indicate that samples produce small gradients outside their target's output row, supporting approximate separation of coordinate supports. This concentration varies across models and checkpoints and is less evident in ResNet-20 at initialization. Nonzero off-diagonal gradients show that the separation is not exact.

E.2 ROW AND COLUMN ORTHOGONALITY

Under Assumption 4, the gradients of different components must have orthogonal row and column spaces. We measure these using normalized pairwise overlaps,

$$R_{ij} = \frac{\|\mathbf{G}_i\mathbf{G}_j^\top\|_F}{\|\mathbf{G}_i\|_F\|\mathbf{G}_j\|_F}, \qquad K_{ij} = \frac{\|\mathbf{G}_i^\top\mathbf{G}_j\|_F}{\|\mathbf{G}_i\|_F\|\mathbf{G}_j\|_F}.$$

Each entry compares two samples. For nonzero gradients, $R_{ij} = 0$ means their row spaces are orthogonal, while $K_{ij} = 0$ means their column spaces are orthogonal. Low between-cluster values in both measurements therefore support the assumption. These overlaps weight directions by gradient magnitude, so self-overlap need not equal one.

Language: Pythia-1B We compare blocks 0, 7, and 15 at initialization (Figure 13) and step 18,000 (Figure 14). Each row/column pair uses a shared DBSCAN ordering based on concatenated row- and column-overlap profiles.

At initialization, Q/K column overlap is generally low, without the pronounced blocks visible in block-7 value, attention-output, and MLP gradients. Row overlap varies more strongly: it is relatively low for several block-0 projections but widespread in middle and deep value and attention-output projections.

Training changes these patterns differently across layer types. Q/K develop more diffuse column overlap, particularly in block 0, whereas value, attention-output, and MLP-up column overlap remains mostly low outside localized groups. Column blocks remain visible in block-15 value and MLP-down gradients. Row overlap is still substantial in many layers, especially block-7 value gradients, while MLP-down gradients generally show less widespread row overlap. There is therefore no uniform improvement in joint orthogonality with either depth or training.

Overall, column overlap is generally lower than row overlap, providing only partial support for Assumption 4. Low overlap throughout a matrix differs from block structure: the former indicates little overlap between individual samples, while the latter indicates shared directions within groups and separation between groups. Neither pattern alone verifies both conditions of the assumption across all parameter values.

Vision: ResNet-20 At initialization, row overlap is substantial throughout the network and particularly high in stages 2 and 3 (Figure 15). Column overlap is generally lower in these deeper stages than in the stem and stage 1, although appreciable between-group overlap remains.

By epoch 30, many stage-2 and stage-3 column panels show lower background overlap, with localized groups remaining visible (Figure 16). The pattern is not uniform: stage-2/block-0/conv1 retains broad column overlap, and early layers remain comparatively diffuse. Row overlap remains substantial, although several deeper layers show lower overlap than at initialization. There is no consistent conv1-versus-conv2 ordering across stages.

The trained classifier shows particularly clear column separation alongside appreciable row overlap. Thus, as in Pythia-1B, the evidence is stronger for column separation than for joint row and column orthogonality.

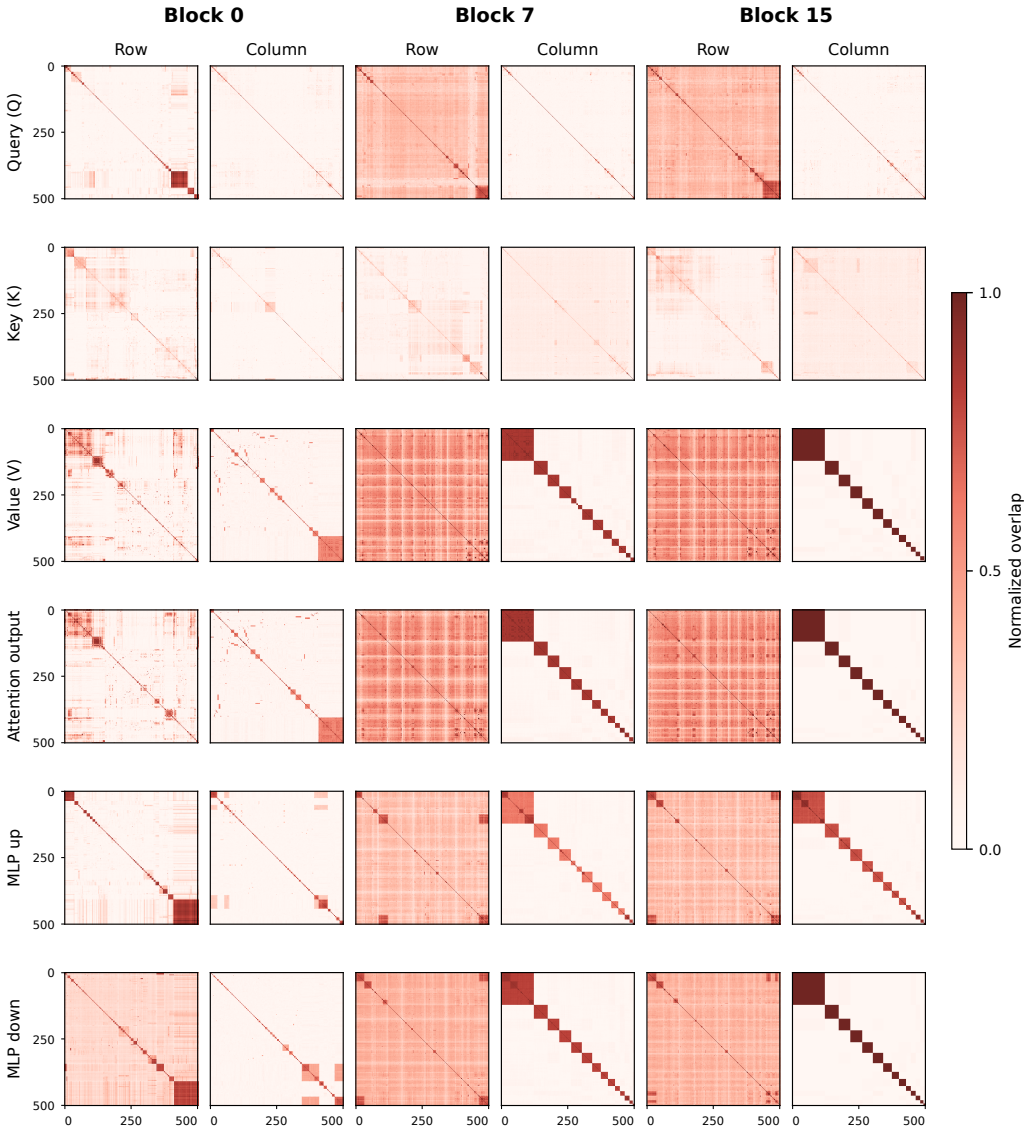

Figure 13: Normalized row and column overlaps across Pythia-1B layers at initialization. Each panel shows 500 samples.

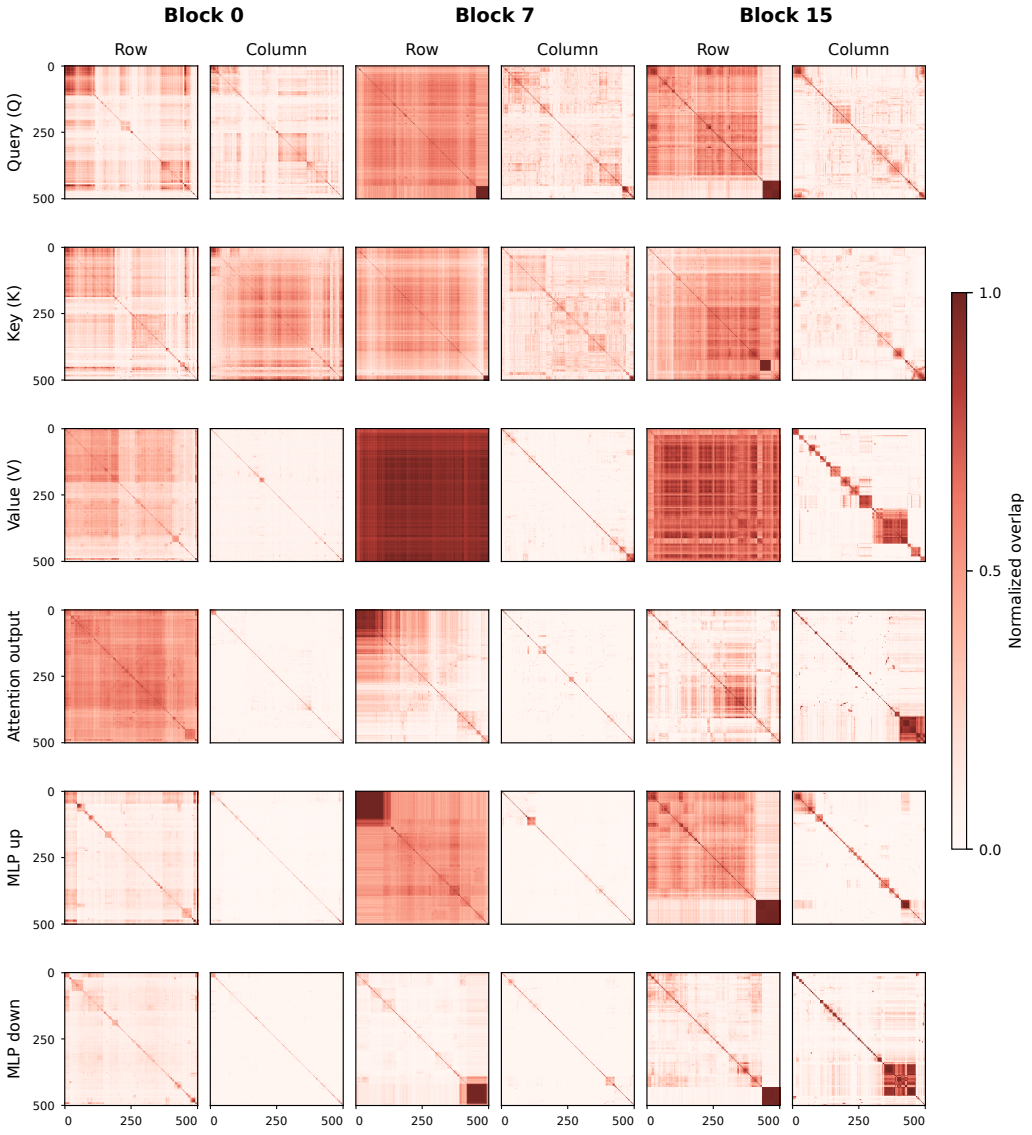

Figure 14: Normalized row and column overlaps across Pythia-1B layers at step 18,000. Each panel shows 500 samples.

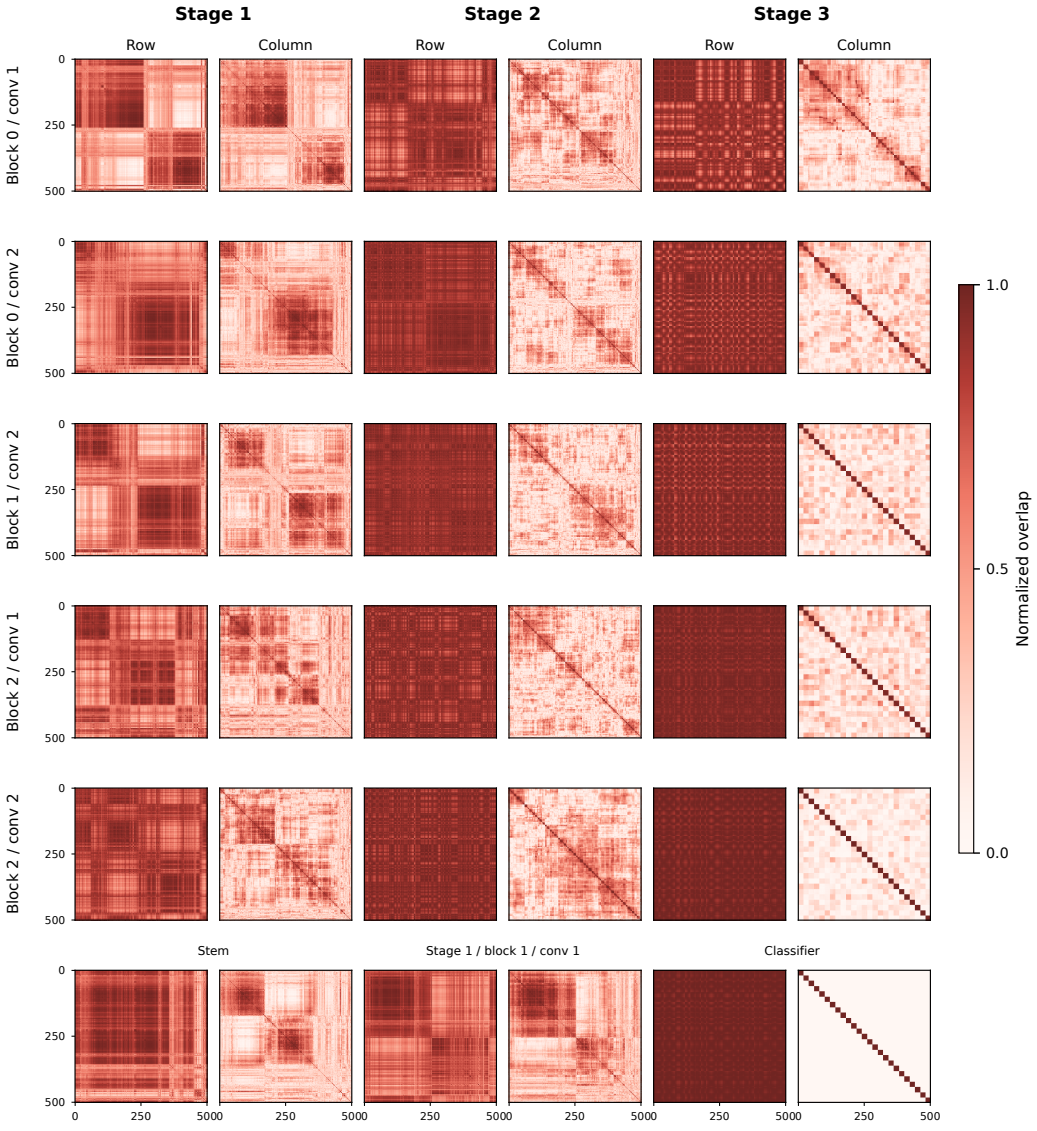

Figure 15: Normalized row and column overlaps across ResNet-20 layers at initialization. Each panel shows 500 samples.

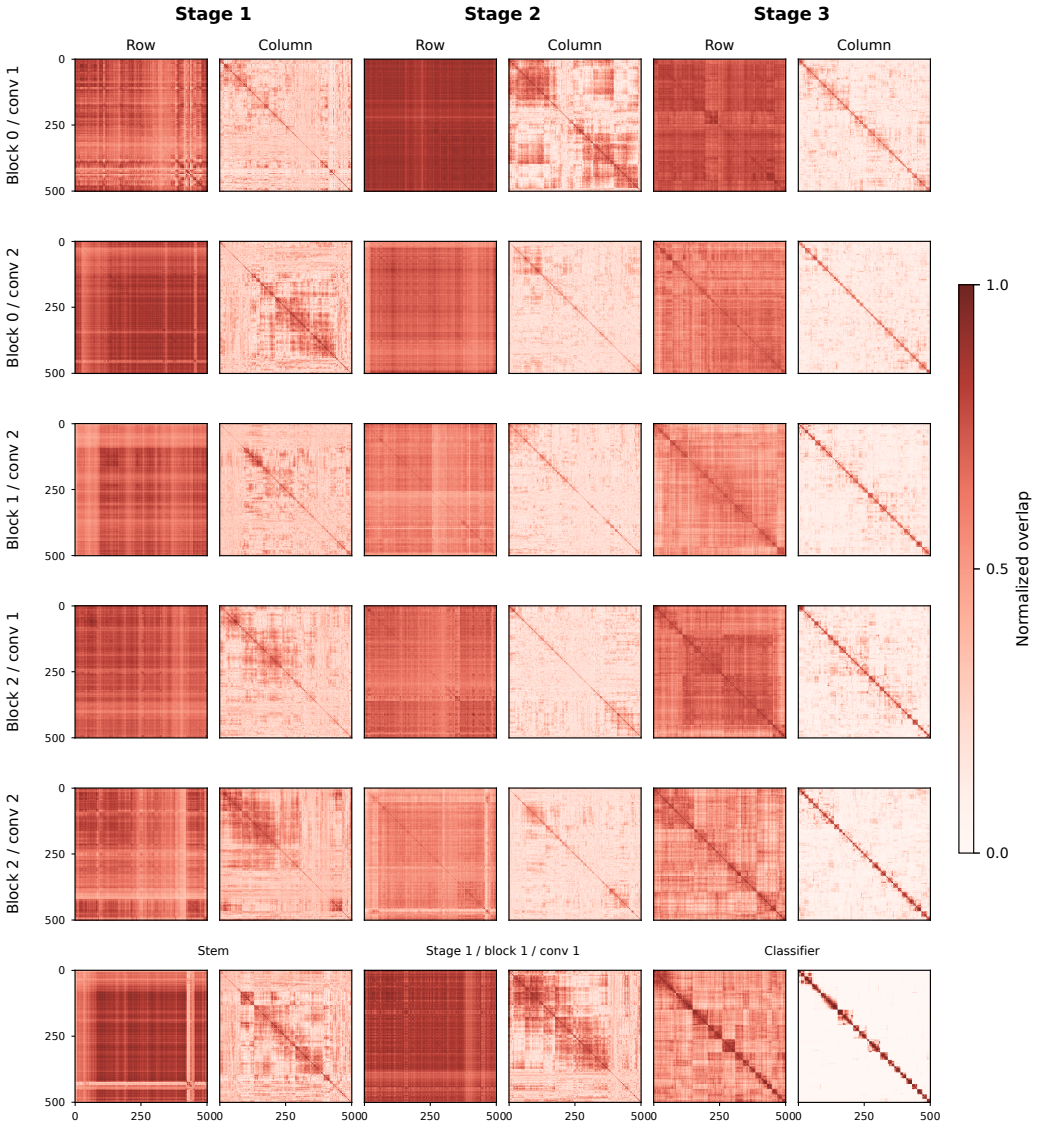

Figure 16: Normalized row and column overlaps across ResNet-20 layers at epoch 30. Each panel shows 500 samples.

F EVIDENCE OF GRADIENT CLUSTERS AND IMBALANCE IN LANGUAGE MODELS

This section discusses results across all layers and training checkpoints for the two language models: the two-layer transformer trained on WikiText-103, and Pythia-1B.

F.1 GRADIENT CLUSTER IMBALANCE IN A TWO-LAYER TRANSFORMER

We show the cluster patterns of the sample Gram matrix for all layers at initialization in Figure 17 and Figure 18 for the first and second module, respectively. After permutation, the sample Gram matrix of the gradients has a block-diagonal structure. Observe that our clustering methodology for the gradients with respect to the Q and K appears somewhat less reliable. The points in the identified clusters are less uniform, and the projection. Whether this is due to a real difference is unclear. A strong cluster structure may exist for those matrices as well, but our heuristic is less suited to those parameters and did not find it. Although our results are not conclusive on the Q, K matrices, it is interesting to note that those coincide with the observations of Wang et al. (2026) that using Muon for those layers is less critical for performance.

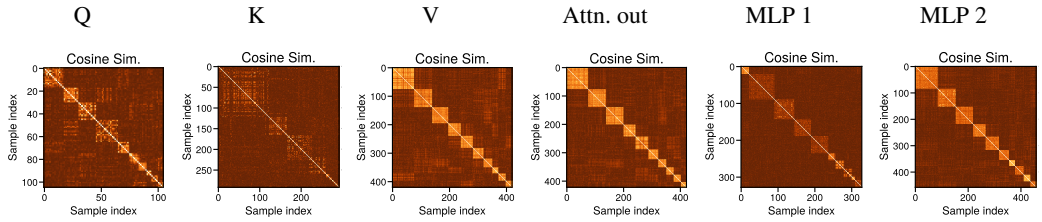

Figure 17: **Two layer transformer, Signal imbalance at initialization: First module.** Gradient-similarity matrices for Q, K, V, attention output, and the two MLP layers, from left to right.

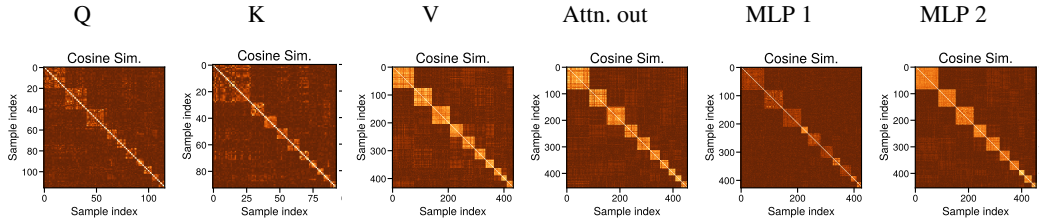

Figure 18: **Two layer transformer, Signal imbalance at initialization: Second module.** Panels follow the same order as Figure 17.

The cluster structure and the semantic meaning of the clusters—what feature samples in a cluster have in common—evolves over time. We focus on two examples to showcase this behavior, the gradients w.r.t. the Q, K matrices of the first module and the gradients w.r.t. the linear layers of the second MLP module. We show the cluster structure in Figure 19 and Figure 20 respectively.

F.2 GRADIENT CLUSTER IMBALANCE IN PYTHIA 1B

Gradient clustering is visible across the analyzed layer types, although its significance varies. Figure 21 illustrates how reordering reveals this structure. In the original sample order, most similarities appear scattered. DBSCAN groups these entries into diagonal blocks of unequal sizes, which are easier to see in the 500-sample crop. This crop highlights the leading groups rather than representing the full distribution of cluster sizes.

Across training time Output-layer gradients show diagonal blocks at every displayed checkpoint, but their sizes and within-block similarities change during training. At initialization, the blocks are sharply separated. At step 1,000, the leading block is smaller, while larger groups reappear

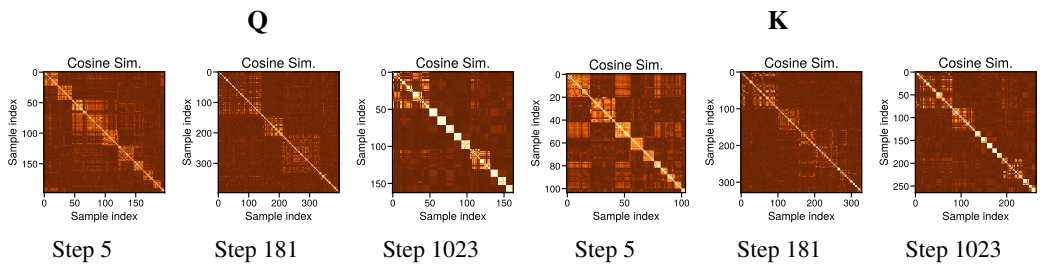

Figure 19: **Evolution of the cluster structure during training for the Q and K matrices in a two-layer transformer.** The first three panels show Q and the last three show K. Clustering becomes stronger after a few optimization steps before weakening again.

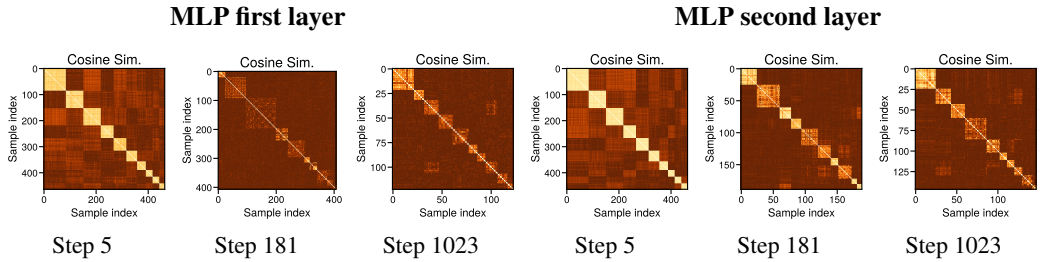

Figure 20: **Evolution of the cluster structure during training for the two linear layers of the last MLP block in a two-layer transformer.** The first three panels show the first layer and the last three show the second layer. Cluster sizes decrease during training.

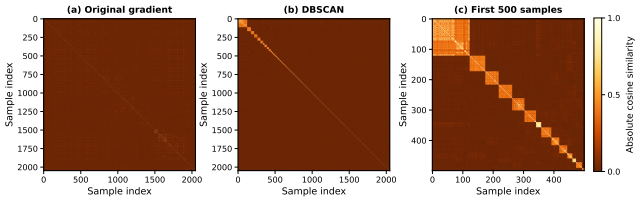

Figure 21: **Reordering reveals unequal gradient clusters, Pythia-1B.** Absolute cosine similarity between per-sample gradients of the block 15 MLP down-projection at initialization. Left: original sample order. Middle: DBSCAN ordering. Right: the first 500 samples after reordering.

at later checkpoints, sometimes with weaker or less uniform similarity. The plots therefore show reorganization rather than a steady increase or decrease in clustering. Since samples are reordered independently at each checkpoint, blocks at the same position need not contain the same samples.

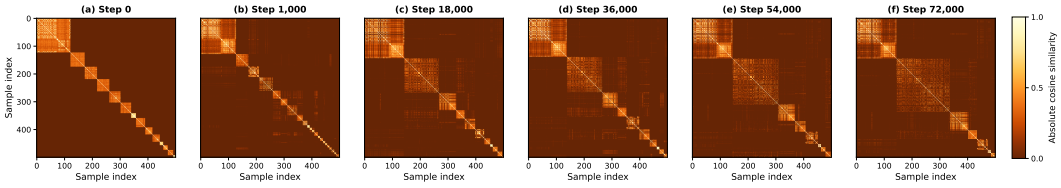

Figure 22: **Evolution of output layer gradient clustering across training steps.** Cosine similarity Gram matrices of per-sample gradients show strong block structure at initialization that progressively reorganizes during training. Samples are reordered independently at each checkpoint using DBSCAN cluster assignments, and each panel shows the first 500 samples after reordering.

Across depth : At initialization, MLP down-projections in Appendix F.2 show a clear difference between block 0 and the later blocks. Block 0 contains smaller groups and more scattered similarities,

while blocks 1, 7, and 15 show a large leading group followed by smaller diagonal blocks. The later blocks look broadly similar, rather than showing a steady increase in cluster size with depth. The attention-output projections at step 72,000 in Appendix F.2 show a less consistent depth trend. Their blocks are generally weaker than those in the initialization MLP example, and block 15 contains several small, bright groups alongside scattered off-diagonal similarities. Thus, clearer clustering in deeper layers is not a uniform pattern across layer types and checkpoints.

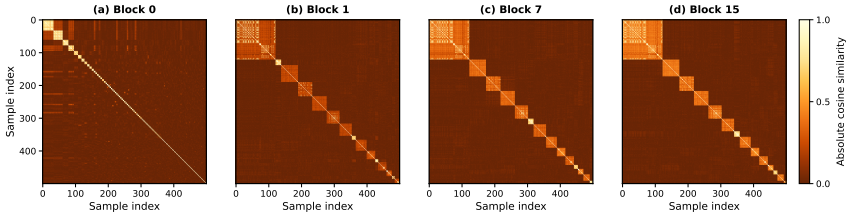

Figure 23: **Per-sample gradient similarities across layer depth.** DBSCAN-reordered similarity matrices for Pythia-1B MLP down-projections at initialization. Cluster structure becomes increasingly block-like across network depth.

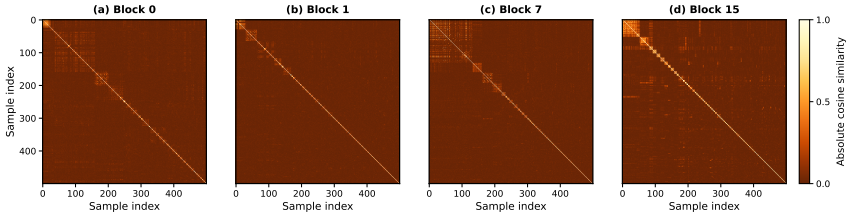

Figure 24: **Per-sample gradient similarities for attention out layer across depth at step 72000**.

Across layer types Figures 25 and 26 compare gradient structure across parameter types. At initialization, value and attention-output projections show clear diagonal blocks in blocks 1, 7, and 15, whereas Q/K projections show little comparable grouping. MLP projections and the output head also exhibit clear blocks, while input-embedding groups are smaller.

At step 72,000, Q/K similarities become more widespread but often remain widespread rather than forming well-separated blocks, except for block-15 K, which shows clearer grouping. Value projections retain stronger blocks in block 15, as do the MLP projections, while attention-output groups are generally smaller and less pronounced. The output head continues to show unequal blocks. Thus, V differs substantially from Q/K, and the relative strength of clustering across parameter types changes during training.

F.3 Semantics of the language model clusters

Two-layer transformer. In the two-layer transformer, gradient clusters reflect different patterns across layers and training. Early in training, clusters in the first block's query projection group short contexts with similar tokens (Table 2), while those in the second block's MLP output projection mainly share the target token (Table 4). By the end of training, query clusters often contain overlapping prefixes of the same sequence, even when their targets differ (Table 3). The MLP clusters become smaller and appear to reflect bigram patterns between the last input token and the target (Table 5). These examples suggest that gradient grouping shifts from individual token identities toward shared context and token co-occurrences during training.

Pythia 1B We examine the three largest DBSCAN clusters from 2,048 prediction samples per analysis. Tables 6 to 12 show the MLP down projections in blocks 0 and 15 and the output head at initialization and step 72,000, together with block 7 attention output at step 72,000.

Training broadens groups beyond identical targets. All nine reported initialization clusters contain a single target-token ID, whereas every reported cluster at step 72,000 contains multiple targets. For

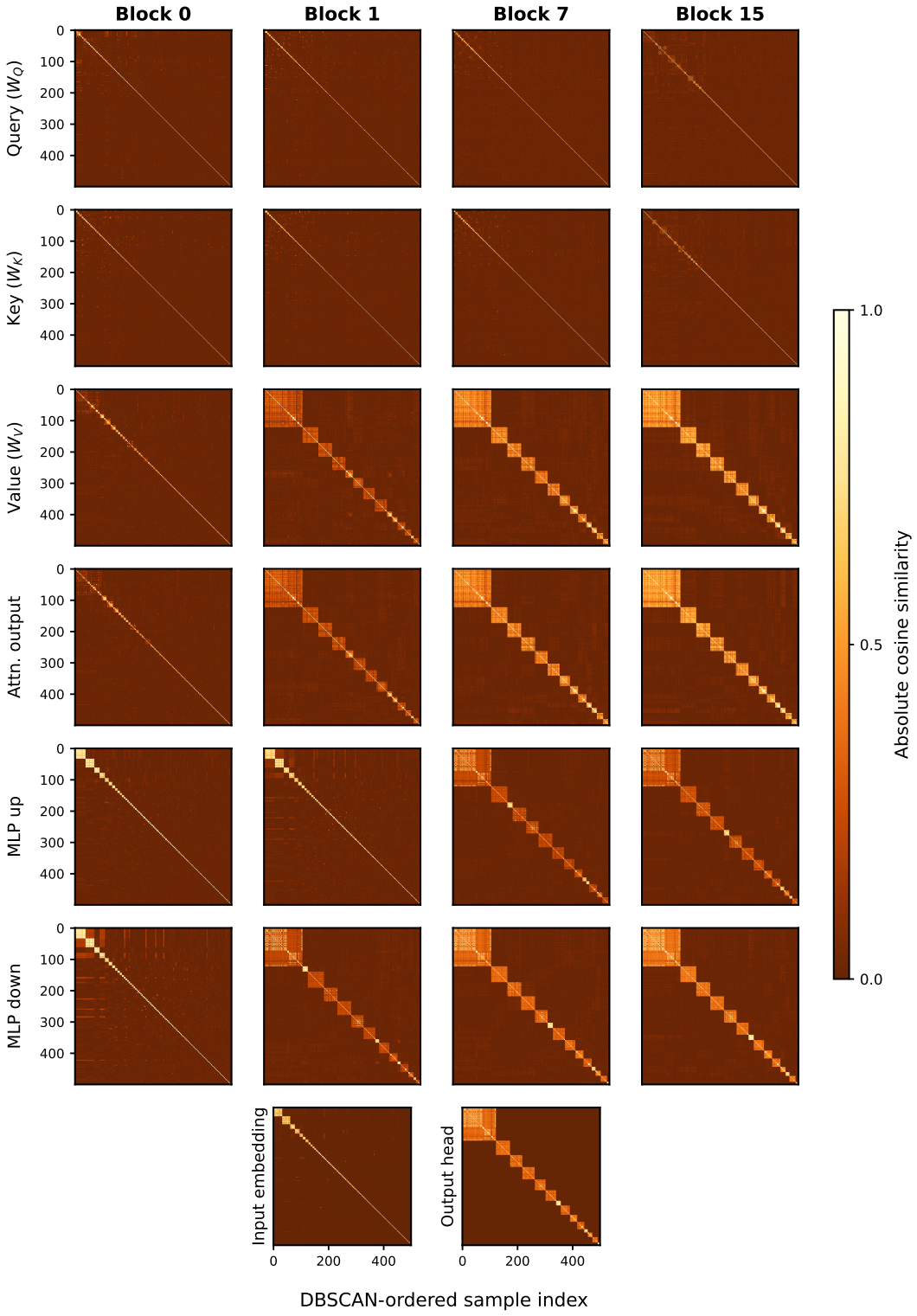

Figure 25: DBSCAN reordered gradient similarity across parameter types and model depth in Pythia-1B at initialization. Each heatmap shows the absolute pairwise cosine similarity cropped at top 500 sample indices, reordered independently using DBSCAN. Columns show representative transformer blocks, while rows show attention projections, attention output, MLP projections, input embeddings, and the output head. All panels share the same color scale from 0 to 1.

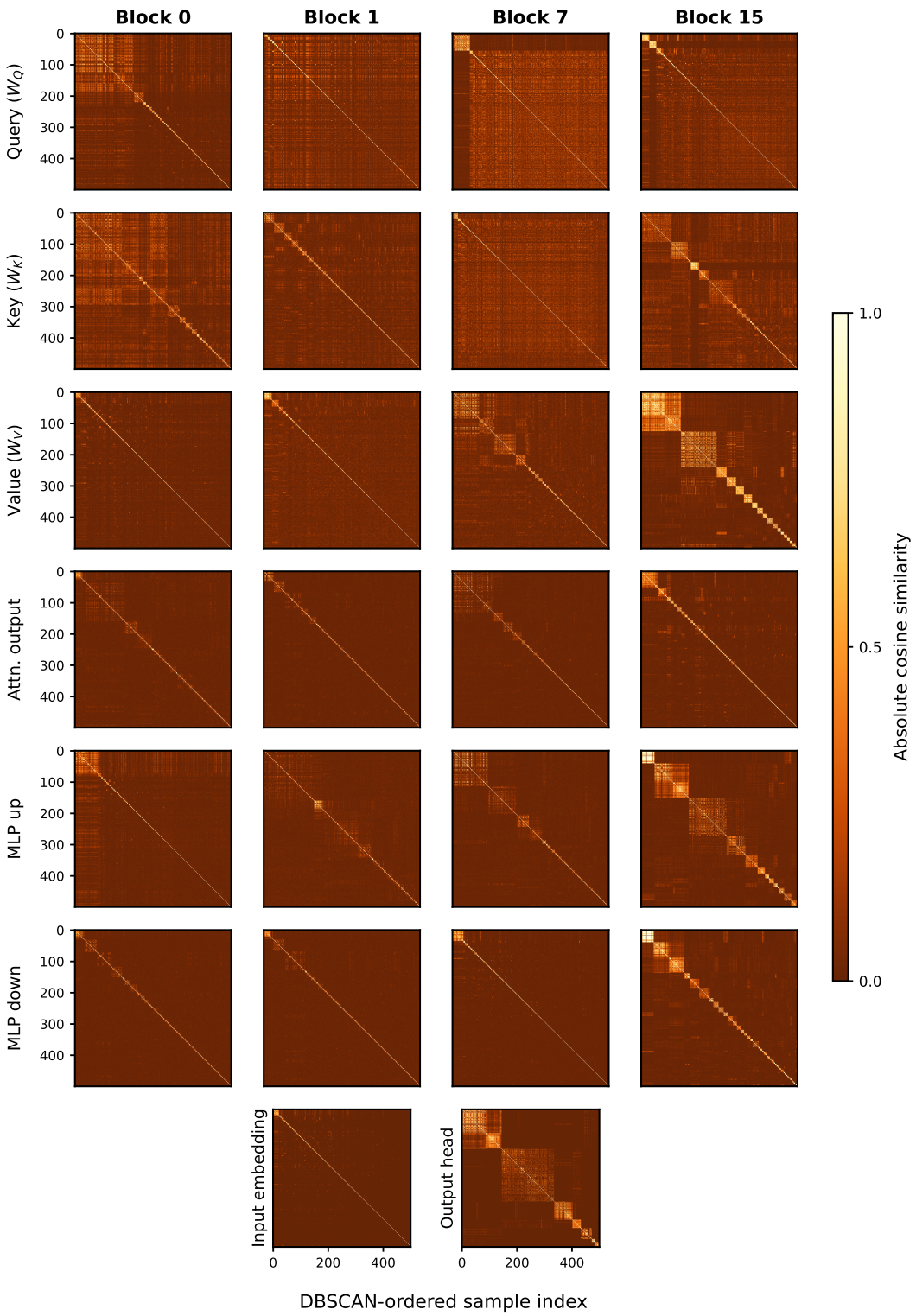

Figure 26: DBSCAN reordered gradient similarity across parameter types and model depth in Pythia-1B at step 72000.

Cluster	Input	Output
1	| ,|	| and|
	| first| ,|	| and|
	| lot| ”| ,|	| but|
	| ,|	| K|
	| ,|	| notably|
	| ,|	| killing|
	| ,|	| Wals|
2	| the|	| 2005|
	| album| .| According| to| the|	| producer|
	| album| .| According| to| the| producer| ,| few| days| after| he| had| finished| the|	| composition|
	|iting| occurs| in| the|	| spring|
	| the|	| population|
	| while| revenue| from| the|	| Federal|
	| R|otten| Tom|atoes| gives| the|	| film|
3	|ite| bed|rock| .|	| Whit|
	| .|	| |
	| .|	| Lind|
	| .|	| |
	|s| .|	| The|
	| .|	| Its|
	| album| .|	| According|
4	| of|	| the|
	| of| the|	| United|
	| of| the| United| States| ’| declaration| of| war| on| 6| April| 1917| .| Tucker| joined| the|	| fleet|
	| a| writer| for| anime| who| had| previously| collaborated| with| Tak|ah|ashi| on| adaptations| of| the|	| first|
	| the| population| of|	| the|
	| the| population| of| the|	| metropolitan|
	| the| population| of| the| metropolitan| area| (| the| administrative| area| of| the|	| City|
5	| ,| and|	| headed|
	| ,| and| headed|	| back|
	| ,| and| headed| back|	| to|
	| first| ,| and|	| for|
	| game| developed| and|	| published|
	|house| and|	| weight|
	| and|	| retained|

Table 2: **Large clusters in Q matrix in the two-layer transformer at the start of training are short contexts with similar tokens.** Text samples from each of the top 5 clusters of the Q matrix of the first module at the start of training.

Cluster	Input	Output
1	| of| the|	| United|
	| of| the| United|	| States|
	| of| the| United| States|	| '|
	| one| million| copies| .|	| In|
	| one| million| copies| .| In|	| an|
	| one| million| copies| .| In| an|	| experience|
	| one| million| copies| .| In| an| experience|	| similar|
2	|is|abeth|	| '|
	|is|abeth| '|	|s|
	|is|abeth| '|s|	| funeral|
	|is|abeth| '|s| funeral|	| ,|
	|is|abeth| '|s| funeral| ,|	| Victoria|
	|is|abeth| '|s| funeral| ,| Victoria|	| removed|
	|is|abeth| '|s| funeral| ,| Victoria| removed|	| her|
3	| most| common| forms|	| include|
	| most| common| forms| include|	| the|
	| most| common| forms| include| the|	| inc|
	| most| common| forms| include| the| inc|	|ense|
	| most| common| forms| include| the| inc|ense|	| stick|
	| most| common| forms| include| the| inc|ense| stick|	| and|
	| most| common| forms| include| the| inc|ense| stick| and| inc|	|ense|
4	|s| offense| again|	| failed|
	|s| offense| again| failed|	| to|
	|s| offense| again| failed| to|	| advance|
	|s| offense| again| failed| to| advance|	| the|
	|s| offense| again| failed| to| advance| the|	| ball|
	|s| offense| again| failed| to| advance| the| ball|	| .|
	|s| offense| again| failed| to| advance| the| ball| .| After|	| a|
5	| .| Lind|er|f|elt| left|	| Mil|
	| .| Lind|er|f|elt| left| Mil|	|w|
	| .| Lind|er|f|elt| left| Mil|w|	|au|
	| .| Lind|er|f|elt| left| Mil|w|au|kee|	| that|
	| .| Lind|er|f|elt| left| Mil|w|au|kee| that| day|	| for|
	| .| Lind|er|f|elt| left| Mil|w|au|kee| that| day| for| a|	| promised|
	| .| Lind|er|f|elt| left| Mil|w|au|kee| that| day| for| a| promised| job|	| at|

Table 3: **Large clusters in Q matrix in the two-layer transformer at the end of training are different predictions tasks on the same sequence, with different context lenghts.** Text samples from each of the top 5 clusters of the Q matrix of the first module at the end of training.

Cluster	Input	Output
1	| they| never| discussed| the| other| fe|ather| @-@| bearing| specimens|	| ,|
	|enced|	| ,|
	|enced| ,| single| @-@| handed|	| ,|
	|ite| bed|rock| .| Whit|em|ore| creek|	| ,|
	|ite| bed|rock| .| Whit|em|ore| creek| ,| a| tribut|ary| of| the| Me|ander| River| (| Tasman|ia|)| ,| flows| past| Whit|em|ore| .| The| creek| is| seasonal|	| ,|
	| .| |\n| The| museum| '|s| hold|ings| number| over| 2| million| rocks|	| ,|
	| .| |\n| The| museum| '|s| hold|ings| number| over| 2| million| rocks| ,| min|erals| ,| fossils|	| ,|
2	|ite| bed|rock|	| .|
	|ite| bed|rock| .| Whit|em|ore| creek| ,| a| tribut|ary| of| the| Me|ander| River| (| Tasman|ia|)| ,| flows| past| Whit|em|ore|	| .|
	| .| |\n| Other| companies| followed| with| their| own| b|ubble| light| versions| and| other| lighting| nov|elt|ies|	| .|
	| .| |\n| Other| companies| followed| with| their| own| b|ubble| light| versions| and| other| lighting| nov|elt|ies| .| In| 1946| ,| S|ylvania| introduced| flu|ores|cent| past|el| lights|	| .|
	|idian| Bank| replaced| M|ell|on| Bank| as| the| lead| ten|ant|	| .|
	| '|m| in| trouble|	| .|
	| ,| and| headed| back| to| New| York| on| 30| September|	| .|
3	| they| never| discussed|	| the|
	|idian| Bank| replaced| M|ell|on| Bank| as|	| the|
	| likely| to| be| a| preliminary| sketch| for| a| now| @-@| unknown| work| .| |\n| =| =| Reception| =| =| |\n| Youth| on| the| Pro|w| ,| and| Ple|asure| at|	| the|
	| of|	| the|
	| brigade| and| ,| during|	| the|
	| in| assets| and| $| 65|4| million| in| l|i|abilities| .| During| bankrupt|cy| ,|	| the|
	|u| ranked| Super| Mario| 64| DS|	| the|
4	| Cl|in|ic| '|s| knowledge|	| and|
	| ,|	| and|
	| near| Gil|ford| .| It| had| been| travelling| from| Ban|bridge| to| |¡unk¿| with| nine| people| on| board| ;| all| were| Catholics|	| and|
	| brigade|	| and|
	| when| Alex|ios| III| married| his| two| daughters| I|rene|	| and|
	| when| Alex|ios| III| married| his| two| daughters| I|rene| and| Anna| to| Alex|ios| Pal|ai|olog|os|	| and|
	| in| assets|	| and|
5	|ite| bed|rock| .| Whit|em|ore| creek| ,| a| tribut|ary|	| of|
	| "| A| Canter|l|ot| W|edding| "| to| viewers| who| may| be| ske|pt|ical| about| the| popularity|	| of|
	| ,| and| headed| back| to| New| York| on| 30| September| .| She| took| on| a| load|	| of|
	| of| the| United| States| '| declaration|	| of|
	| brigade| and| ,| during| the| Valley| Campaign|s|	| of|
	| brigade| and| ,| during| the| Valley| Campaign|s| of| 1864| he| was| briefly| placed| in| command|	| of|
	| brigade| and| ,| during| the| Valley| Campaign|s| of| 1864| he| was| briefly| placed| in| command| of| a| division|	| of|

Table 4: **Large clusters in the last linear layer in the two-layer transformer at the start of training depend on the output token.** Text samples from each of the top 5 clusters.

Cluster	Input	Output
1	| .| |	|\n|
	| .| |	|\n|
	| '|m| in| trouble| .| '| "| |	|\n|
	| likely| to| be| a| preliminary| sketch| for| a| now| @-@| unknown| work| .| |	|\n|
	|u| ranked| Super| Mario| 64| DS| the| 29|th| most| wanted| title| .| |	|\n|
	| 22| was| then| redesign|ated| as| NY| 164| .| |	|\n|
	| have| been| familiar| .| |	|\n|
2	| .|	| |
	| "| A| Canter|l|ot| W|edding| "| to| viewers| who| may| be| ske|pt|ical| about| the| popularity| of| the| series| .|	| |
	| .| Lind|er|f|elt| left| Mil|w|au|kee| that| day| for| a| promised| job| at| the| Library| Bureau| in| Boston| .|	| |
	| .|	| |
	| '|m| in| trouble| .| '| "|	| |
	| 22| was| then| redesign|ated| as| NY| 164| .| |\n| =| =| Major| intersections| =| =| |\n| The| entire| route| is| in| Pat|ters|on| ,| Put|nam| County| .|	| |
	| have| been| familiar| .|	| |
3	| Cl|in|ic| '|	|s|
	| .| |\n| The| museum| '|	|s|
	| of| the| United| States| '|	| declaration|
	| small| proportion| of| their| surface| ,| but| most| or| all| of| Hy|ak|ut|ake| '|	|s|
	|ounded| by| C|att|ell| Road| ,| Cov|entry| Road| ,| T|ilton| Road| ,| Gar|rison| Lane| and| the| railway| ,| and| near| St| Andrew| '|	|s|
	| lot| "| ,| but| disliked| Mon|a| '|	|s|
	| eastern| reaches| of| the| N|esc|op|eck| Creek| watershed| are| near| the| border| of| the| App|al|ach|ian| Plate|au| region| .| |\n| N|esc|op|eck| Creek| '|	|s|
4	| of|	| the|
	| brigade| and| ,| during|	| the|
	| a| massive| flooding| of|	| the|
	| first| ,| and| for| years| the| only| ,| city| in|	| the|
	| Roman| Catholic| Church| on| Malta| .| Within| a| week| ,| Bon|ap|arte| had| res|up|pl|ied| his| ships| ,| and| on| 19| June| ,| his| fleet| departed| for| Alexandria| in|	| the|
	| have| been| familiar| .| |\n| Don|ne| '|s| cart|ographic| references| in|	| the|
	|iting| occurs| in|	| the|
5	| .| |\n|	| The|
	| likely| to| be| a| preliminary| sketch| for| a| now| @-@| unknown| work| .| |\n| =|	| =|
	| 22| was| then| redesign|ated| as| NY| 164| .| |\n| =|	| =|
	|nder| uncertainty| .| |\n|	| =|
	|nder| uncertainty| .| |\n| =|	| =|
	| Person| of| the| Year| recognition| in| 2012| .| |\n|	| =|
	| Person| of| the| Year| recognition| in| 2012| .| |\n| =| =|	| Early|

Table 5: **Large clusters in the last linear layer in the two-layer transformer at the end of training depend on bigrams between the last token of the input sequence and the output token.** Text samples from each of the top 5 clusters.

example, the largest trained output-head cluster contains 190 samples (9.28%) across 11 targets, including punctuation and common words. Newline-dominated groups remain prominent.

Target mixing varies across depth. At step 72,000, the leading shallow-MLP clusters in block 0 contain 11-28 distinct targets, with the most frequent target accounting for 17-31% of each cluster. The deep-MLP clusters in block 15 remain more concentrated, with 3-8 targets and dominant-target shares of 73-85%. Thus, the deeper MLP shows stronger target dependence in this comparison.

Mixed groups can share contextual patterns. Both block 0 MLP down and block 7 attention output contain HTML-related groups with targets such as `>`, `span`, and `li`. These examples suggest that shared gradient responses can reflect formatting or context, rather than target identity alone.

These observations concern selected leading clusters, fitted independently for each layer and checkpoint. They do not establish a unique semantic meaning for every cluster or a uniform increase in frequency imbalance during training.

Target composition	Size (%)	Context → target
[newline] (35); 1 distinct target	1.71	… ist parties won a slim majority in a regional election. [newline] → **[newline]** … in self-imposed exile, or another EU country. [newline] → **[newline]** ependence parties are best placed to form a government. [newline] → **[newline]**
[newline] (28); 1 distinct target	1.37	… Minister Mariano Rajoy later appeared to reject the idea. → **[newline]** … would have to take up their post in Catalonia itself. → **[newline]** … onia to remain a semi-autonomous part of Spain. → **[newline]**
[space] (17); 1 distinct target	0.83	… : #7d868c; [newline] *{ [newline] → **[space]** … webkit-box-sizing: border-box; [newline] → **[space]** … moz-box-sizing: border-box; [newline] → **[space]**

Table 6: Pythia-1B, Block 0 MLP down, step 0. Three largest DBSCAN gradient clusters.

Target composition	Size (%)	Context → target
[newline] (123); 1 distinct target	6.01	… Published duration 22 December 2017 Related Topics Catalonia independence protests → **[newline]** … the time for dialogue," said Carles Puigdemont → **[newline]** … ist parties won a slim majority in a regional election. [newline] → **[newline]**
. (51); 1 distinct target	2.49	… separatist parties won a slim majority in a regional election → **.** … is living in self-imposed exile, or another EU country → **.** … [newline] [newline] The Citizens party is now the region's biggest party → **.**
the (44); 1 distinct target	2.15	"Now is → **the** [newline] He said he would hold talks with the head of → **the** … winner of Thursday's election was Inés Arrimadas, → **the**

Table 7: Pythia-1B, Block 15 MLP down, step 0. Three largest DBSCAN gradient clusters.

Target composition	Size (%)	Context → target
[newline] (123); 1 distinct target	6.01	… Published duration 22 December 2017 Related Topics Catalonia independence protests → **[newline]** … the time for dialogue," said Carles Puigdemont → **[newline]** … ist parties won a slim majority in a regional election. [newline] → **[newline]**
. (51); 1 distinct target	2.49	… separatist parties won a slim majority in a regional election → **.** … is living in self-imposed exile, or another EU country → **.** … [newline] [newline] The Citizens party is now the region's biggest party → **.**
the (44); 1 distinct target	2.15	"Now is → **the** [newline] He said he would hold talks with the head of → **the** … winner of Thursday's election was Inés Arrimadas, → **the**

Table 8: Pythia-1B, Output head, step 0. Three largest DBSCAN gradient clusters.

Target composition	Size (%)	Context → target
[newline] (14); The (4); , (2); 28 distinct targets	2.20	… would have to take up their post in Catalonia itself. → **[newline]** … onia to remain a semi-autonomous part of Spain. → **[newline]** … remain a semi-autonomous part of Spain. [newline] [newline] → **The**
>(7); span (7); li (6); 11 distinct target	2.00	01 → </ 01</ 01</li → >
x (8); ellar (5); able (4); 21 distinct target	1.76	cryptocurrency remittance provider, has → **partnered** . [newline] [newline] The surprising collaboration will → **see** . [newline] [newline] The surprising collaboration will see Wire → **x**

Table 9: Pythia-1B, Block 0 MLP down, step 72,000. Three largest DBSCAN gradient clusters.

Target composition	Size (%)	Context → target
[newline] (83); " (5); [newline][newline] (5); 7 distinct targets	4.79	… Minister Mariano Rajoy later appeared to reject the idea. → **[newline]** … would have to take up their post in Catalonia itself. → **[newline]** … onia to remain a semi-autonomous part of Spain. → **[newline]**
[newline] (32); He (2); In (1); 8 distinct targets	1.95	… ist parties won a slim majority in a regional election. [newline] → **[newline]** … in self-imposed exile, or another EU country. [newline] → **[newline]** [newline] → **He**
the (24); Catal (7); Spain (2); 3 distinct targets	1.61	[newline] He said he would hold talks with the head of → **the** … wants Catalonia to remain a semi-autonomous part of → **Spain** on Friday. "This is the wish of → **the**

Table 10: Pythia-1B, Block 15 MLP down, step 72,000. Three largest DBSCAN gradient clusters.

Target composition	Size (%)	Context → target
[newline] (36); and (8); [space] (6); 61 distinct target	6.49	called → **for** … ist parties won a slim majority in a regional election. [newline] → **[newline]** he → **is**
> (7); span (7); li (6); 7 distinct targets	1.51	<li → > 01</li → > … >01 [newline] < → li
[newline] (22); ” (1); In (1); 8 distinct targets	1.42	… Minister Mariano Rajoy later appeared to reject the idea. → **[newline]** … would have to take up their post in Catalonia itself. → **[newline]** … onia to remain a semi-autonomous part of Spain. → **[newline]**

Table 11: Pythia-1B, Block 7 attention output, step 72,000. Three largest DBSCAN gradient clusters.

Target composition	Size (%)	Context → target
. (46); , (39); in (36); 11 distinct target	9.28	… Puigdemont [newline] [newline] Catalonia’s ousted leader → **,** … talks with Spain after separatist parties won a slim majority → **in** … separatist parties won a slim majority in a regional election → **.**
[newline] (123); [newline][newline] (6); ” (5); 10 distinct targets	7.08	… Published duration 22 December 2017 Related Topics Catalonia independence protests → **[newline]** … the time for dialogue,” said Carles Puigdemont → **[newline]** … ist parties won a slim majority in a regional election. [newline] → **[newline]**
the (44); Catal (15); Spain (6); 4 distinct targets	3.22	… for talks with Spain Published duration 22 December 2017 Related Topics → **Catal** ”Now is → **the** called for new talks with → **Spain**

Table 12: Pythia-1B, Output head, step 72,000. Three largest DBSCAN gradient clusters.

G CLUSTERING AND IMBALANCE IN VISION

This section discusses results across all layers and training checkpoints for the vision models, Resnet20, trained on CIFAR100.

G.1 CLUSTERING ACROSS DEPTH AND TRAINING

Gradient grouping is generally clearer in deeper layers, although the pattern is not uniform across the network. At epoch 30, the stage-2 and stage-3 layers in Figure 27 show more distinct diagonal groups than the corresponding stage-1 layer. The broader comparison in Figure 29 also contains exceptions: some earlier layers show substantial structure, while some deeper layers have larger, more diffuse groups. The classifier generally shows cleaner separation between groups than the intermediate convolutions.

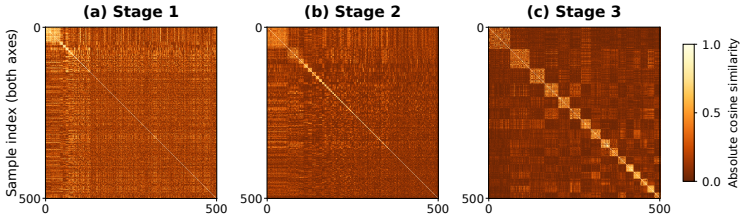

Figure 27: **Gradient clustering varies across depth.** Absolute cosine similarity of per-sample gradients for the second convolution in the first block of each stage at epoch 30. Stages 2 and 3 show clearer diagonal groups than stage 1 in this comparison. Samples are reordered using DBSCAN.

Some grouping is already visible at initialization, particularly in the classifier and deeper layers. In the deep convolution shown in Figure 28, the initial matrix has substantial similarity both within and between groups. During training, more distinct diagonal blocks emerge against a darker background, suggesting greater separation between some gradient groups.

This change is not a steady increase in cluster size or imbalance. Groups can become larger or less dominant at different checkpoints, and the pattern varies by layer. The summary in Figure 29 shows both the emergence of unequal blocks and the variation in their strength across the network.

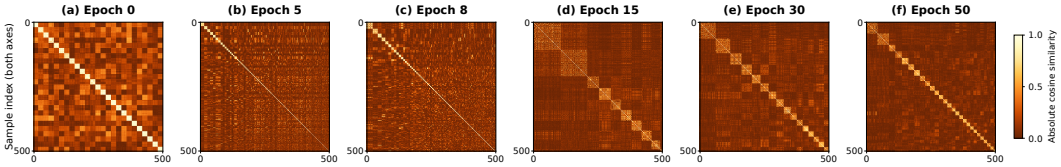

Figure 28: **Training changes the separation of gradient groups.** Absolute cosine similarity for the second convolution in the first block of stage 3 at different checkpoints. Later matrices show more distinct diagonal groups in some checkpoints, but cluster size and separation do not change monotonically. Each matrix is independently reordered using DBSCAN.

G.2 VISUAL AND SEMANTIC PATTERNS WITHIN CLUSTERS

We inspect images from the largest gradient clusters to understand which samples produce shared gradient directions. Figures 30 and 31 show examples from the classifier and a deep convolutional layer, respectively.

At initialization, each of the two highlighted classifier clusters contains 20 images from a single class, matching the class-balanced sampling. These initial groups therefore need not reflect learned visual features. At later checkpoints, the highlighted classifier clusters contain images from several classes, including related categories such as streetcars, buses, and trains, or whales, sharks, and dolphins. Other examples appear to share visual properties across categories, such as color, shape, texture, or background.

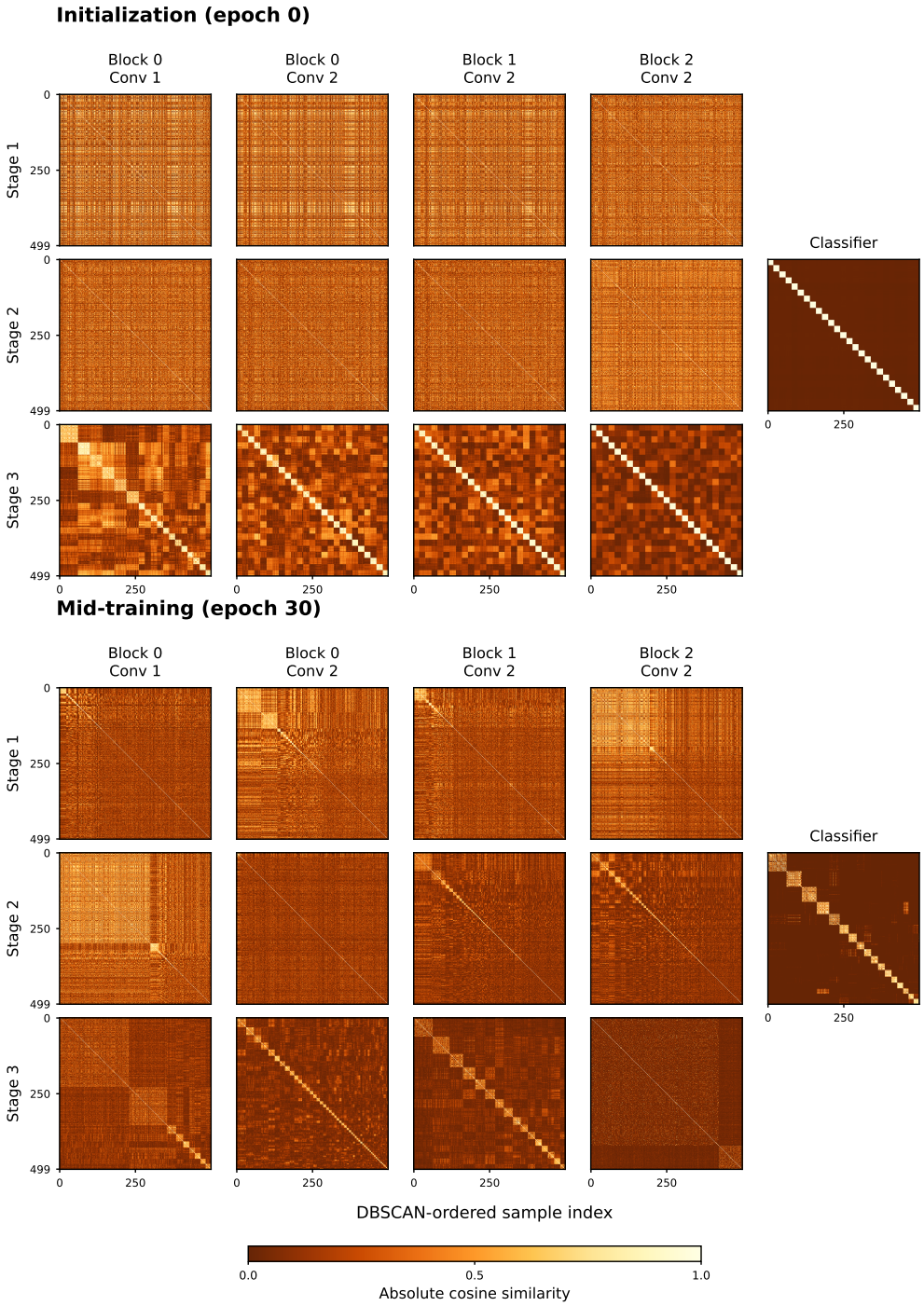

Figure 29: **Clustering and imbalance vary across ResNet-20 layers and training checkpoints.** Absolute cosine similarity matrices reordered using DBSCAN. The broader comparison shows that clearer grouping in deeper layers is a tendency rather than a uniform ordering across depth or training.

These examples show that gradient groups need not coincide with class labels: balanced class counts can coexist with unequal groups of samples that produce similar gradients. Shared visual properties offer a possible explanation for some groups. Absolute cosine similarity includes both aligned and opposite gradients, and the displayed images represent only the highest-membership samples. The figures therefore illustrate possible visual and semantic associations, rather than assigning a unique meaning to every cluster.

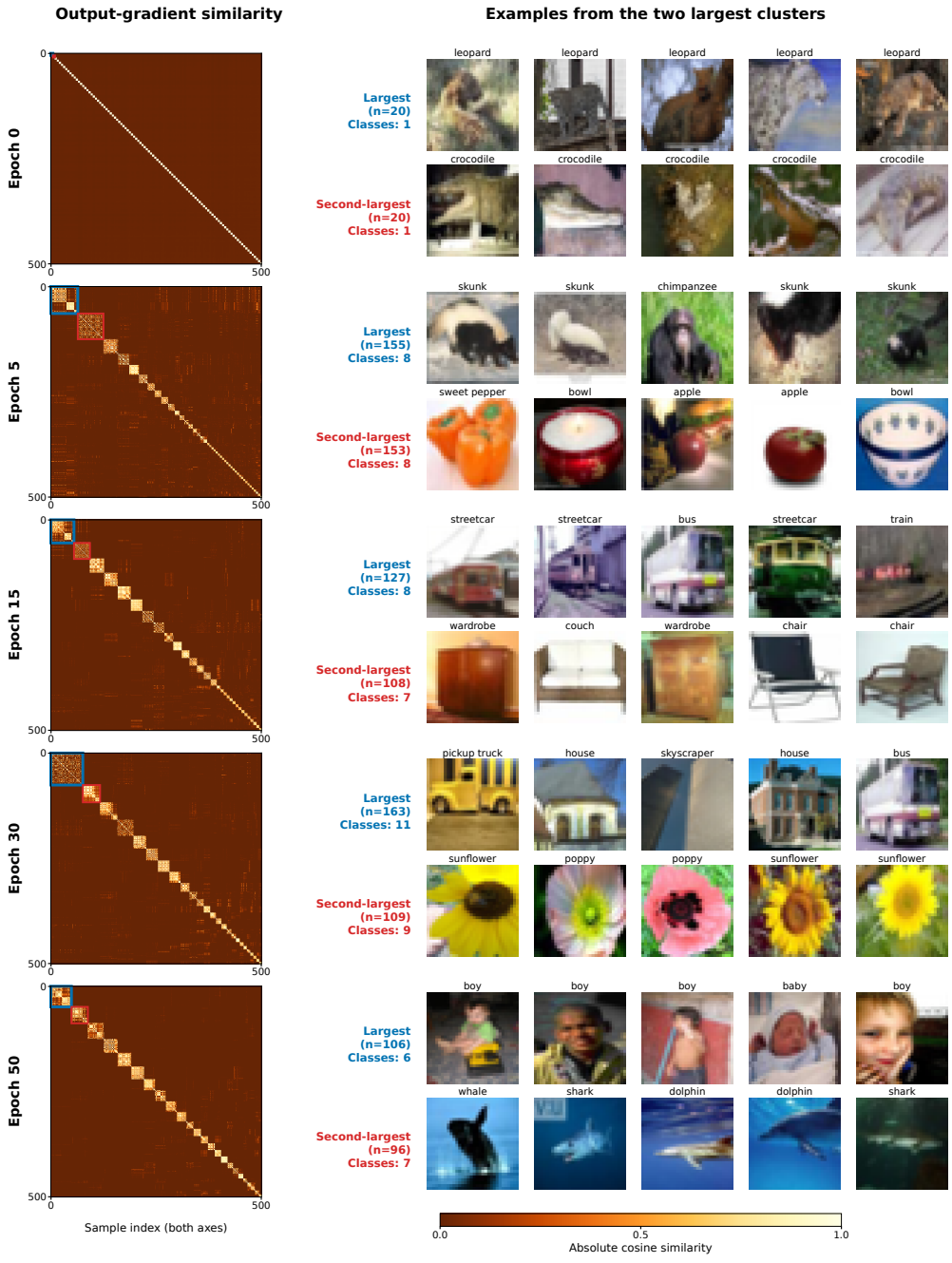

Figure 30: **Classifier gradient clusters extend beyond individual classes during training.** Similarity matrices and image examples from the two largest clusters at each checkpoint. Each highlighted initialization cluster contains images from one class, whereas trained clusters span multiple classes. The five highest-membership images per HDBSCAN cluster are shown with their ground-truth labels; reported counts describe the full clusters.

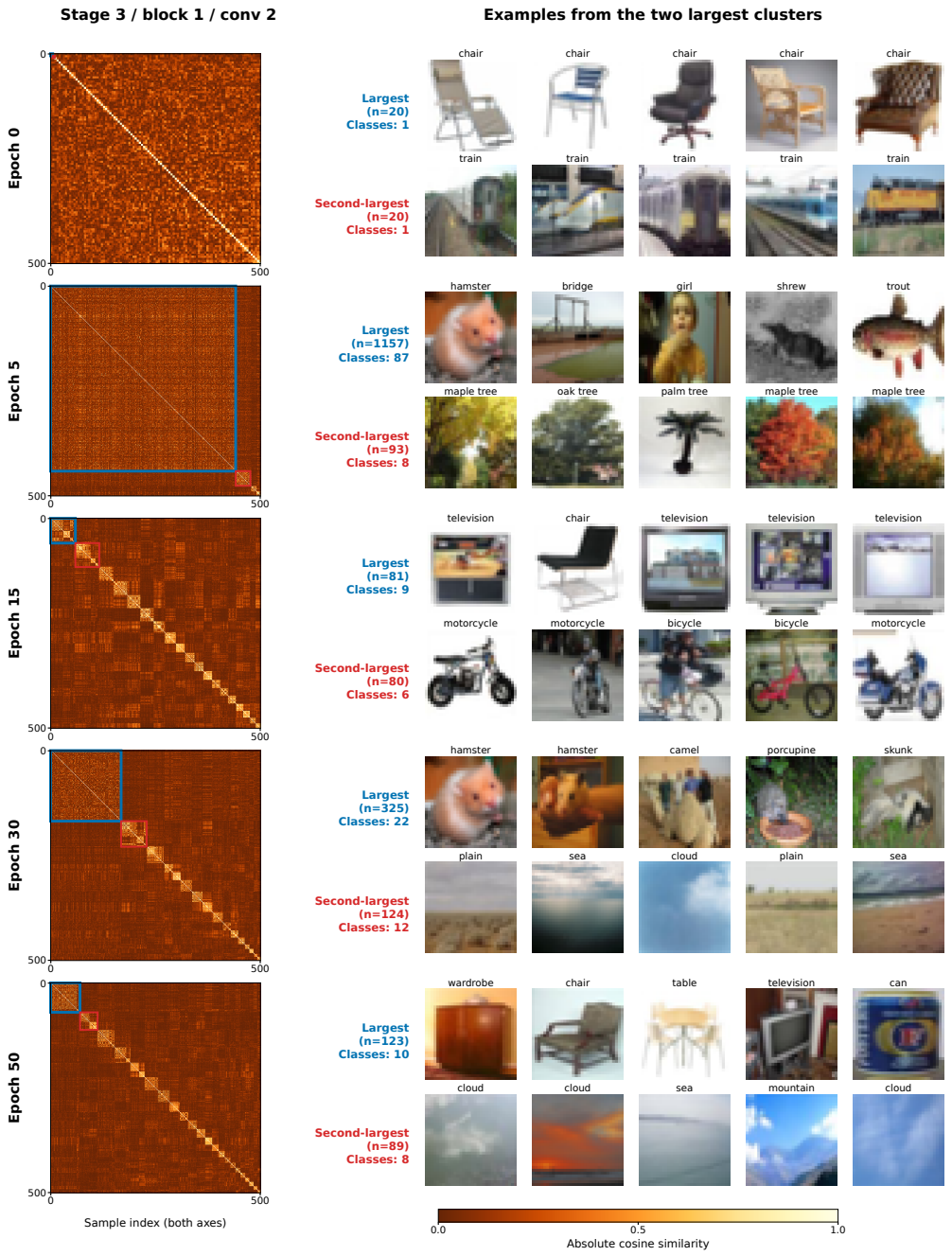

Figure 31: **Deep convolutional gradient clusters contain varied cross-class groups.** Similarity matrices and image examples for the second convolution in the first block of stage 3. Some examples share apparent visual or semantic properties, while others belong to broad, heterogeneous groups. Each checkpoint is clustered independently. The five highest-membership images from each highlighted HDBSCAN cluster are shown.