Neural Networks Loss Landscape Convergence in Different Low-Dimensional Spaces

Tem Nikitin
Moscow Institute of Physics and Technology
Moscow, Russia
nikitin.artem.a@phystech.su

Nikita Kiselev
Moscow Institute of Physics and Technology
Moscow, Russia
kiselev.ns@phystech.su

Andrey Grabovoy
Moscow Institute of Physics and Technology
Moscow, Russia
grabovoy.av@phystech.su

Vladislav Meshkov
Moscow Institute of Physics and Technology
Moscow, Russia
meshkov.ns@phystech.su

Abstract

Understanding how the loss landscape of neural networks evolves as the training set size increases is crucial for optimizing performance. It is well known that larger datasets can alter the shape of this high-dimensional landscape. But the exact point at which additional data no longer brings substantial changes remains underexplored.

In this paper, we examine neural network models and show that their loss landscapes begin to stabilize once the training set grows beyond a certain threshold, revealing a connection between dataset size and the geometry of the loss surface. To understand this phenomenon, we propose a method that projects the full parameter space onto a low-dimensional subspace derived from top eigenvectors of the Hessian. That provides a more interpretable view of how the loss surface in the vicinity of local minima behaves as more data become available. We use different sampling strategies, applying Monte-Carlo estimation to capture the structure of this reduced loss landscape more precisely.

We validate our insights through experiments on image classification tasks, demonstrating that this low-dimensional analysis can reveal when the landscape effectively settles, and thus helps determine a minimum viable dataset size. Our findings shed light on the relationship between dataset scale and optimization geometry, and suggest practical strategies for balancing computational costs with the benefits of additional training data.

Keywords: Neural networks, Loss landscape, Low-dimensional subspace, Hessian eigenvectors, Monte Carlo estimation, Dataset size threshold.

1 Introduction

Neural networks have become irreplaceable in various aspects of life and have a lot of applications now (e.g. image classification, language models, recommender systems, etc.). However, as datasets and models grow larger and larger, we get higher accuracy and breakthrough results. However some problems connected with computation resources have appeared. Sometimes models have so many parameters so we need time we cannot provide for their training. Some attempts to explore and compare optimization methods have already been made [1]. The core problem of all these solutions hides in their locality, as the size of the neural network and training dataset remains unchanged.

In this paper we explore when adding more data no longer brings significant improvements in network performance and to explain obtained results in terms of the curvature and shape of the neural network's loss landscape.

Our research on the loss landscape of neural networks. Specifically, how it changes while adding new samples to the training dataset. There remains a problem: processing of large datasets requires a lot of time. We need to determine when increasing dataset size stops reshaping this loss landscape for general neural networks. But the question of a minimum viable dataset size — a threshold beyond which further data bring insignificant changes — remains underexplored. Moreover, connecting such a threshold to generalization capabilities [2] will reduce computation requirements as we show that there is an upper bound of "active" dataset samples.

We suggest using the Hessian of the loss function to calculate its close approximation. However, the computation of all Hessian requires a bunch of time, so we consider a projection of neural network parameters space to a certain subspace. We place the following tasks to the front:

1. Constructing a Hessian-based projection of the loss landscape which finds main curvature directions (top eigenvectors) in parameter space.
2. Using different sampling methods, identifying at which dataset size the shape of the loss function stabilizes.
3. We apply visualization methods to characterize the loss landscape next to the local minima.
4. Providing applicable theoretical criteria to determine when adding more samples has minimal effect. We refer to The Matrix Cookbook [3] as a hint for matrix operations.

The proposed solution is novel as it links Hessian estimation to the idea of a minimal dataset threshold. This enables advantages such as more cost-effective dataset collection and a clearer understanding of how dataset size interacts with the geometry of the loss function. And applying recent methods for low-rank Hessian estimation can speed up and solve the problem of high computation cost of top eigenvalues for very large networks.

We validate our analysis results on well-known image classification tasks like MNIST [4] and Fashion-MNIST [5]. By connecting practical results to theoretical bounds, we offer a method for identifying a computation-efficiency method that balances performance with accuracy gains.

Contributions. Our contributions can be summarized as follows:

- We present a Hessian-based approach that uses a projection into a low-dimensional subspace of top eigenvectors to find the critical sufficient dataset size.
- We demonstrate the validity of our theoretical framework through empirical studies on MNIST and Fashion-MNIST, incrementally increasing dataset size until additional data makes an impact to the loss function curvature.
- We highlight the implications of our findings for practical data collection strategies, showing how the detection of a sufficient dataset size can reduce computation time.

Outline. The rest of the paper is organized as follows. Section 2 divides existing results into some topics, highlighting their key contributions and findings. Section 3 considers general notation and some preliminary calculations. In Section 4, we provide theoretical bounds on the hessian and losses difference norms. Empirical study of the obtained results is given in Section 5. We summarize and present the results in Sections 6 and 7. Additional experiments and proofs of theorems are included in the Appendix A.

2 Related Work

Hessian-based analysis. The landscape of loss functions has been explored from various perspectives in the literature. Methods connected with Hessian are central to understanding the convergence process and loss function landscape [6]. Recent works observed that the Hessian of some models tends to have a low effective rank, with only a small quantity of eigenvalue differences from zero. Empirical explorations from [7] show that the Hessian rank is often very low in the vicinity of local minima, highlighting a significantly smaller "active subspace". Prior work investigating how large eigenvalues emerge during training [8].

Some results have already been reached for fully connected and convolutional architectures [9], but they need generalization.

Loss Landscape Geometry. Other recent researches aimed at how dataset size affects both performance and the geometry of loss function landscapes. In the assumption of enough training samples, neural networks often reach smoother and flatter minima with better generalization [2]. However, collecting large datasets is expensive, as well as the computation requirements for them, raising questions about the most effective ratio between dataset and model sizes [10]. As understanding the landscape of loss functions is key to the solution, prior work has looked at visualizing loss surfaces [11].

3 Preliminaries

3.1 General notation

In this section, we introduce the general notation used in the rest of the paper and the basic assumptions.

We consider a K-label classification problem. So, let's consider $p(y|\mathbf{x})$ a conditional probability, which maps unobserved variable $\mathbf{x} \in \mathbf{X}$ to the corresponding output $y \in \mathbf{y}$. We consider $\mathbf{y}$ is a subspace (or same space) as $\mathbb{R}^K$. Let $f_{\mathbf{w}}(\cdot)$ be a neural network with a list of parameters $\mathbf{w}$. Let Ω be a space of parameters ($\mathbf{w} \in \Omega$).

Let

$$\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid i = 1, \dots, m\}$$

be a given dataset of size m consists of i.i.d. samples. Let $\ell(\cdot, \cdot)$ be a given loss function (e.g. cross-entropy) where first argument refers to neural network's result and the second argument refers to the true answer. To simplify, define:

$$\ell_i(\mathbf{w}) := \ell(f_{\mathbf{w}}(\mathbf{x}_i), y_i).$$

Definition 1. *The empirical loss function for the first k elements is:*

$$\mathcal{L}_k(\mathbf{w}) = \frac{1}{M} \sum_{1}^{k} \ell_i(\mathbf{w}), \ \mathcal{L}(\mathbf{w}) := \mathcal{L}_m(\mathbf{w}).$$

Thus, the difference between losses for neighbouring samplem sizes is:

$$\mathcal{L}_k(\mathbf{w}) - \mathcal{L}_{k-1}(\mathbf{w}) = \frac{\ell_k(\mathbf{w}) - \mathcal{L}_{k-1}(\mathbf{w})}{k}.$$

Definition 2. *Let the Hessian of $\mathcal{L}_k(\mathbf{w})$ be:*

$$\mathbf{H}_k(\mathbf{w}) = \nabla^2_{\mathbf{w}} \mathcal{L}_k(\mathbf{w}) = \frac{1}{k} \sum_{1}^{k} \nabla^2_{\mathbf{w}} \ell_i(\mathbf{w}).$$

Definition 3. *To calculate the overall loss landscape changing, one has to integrate the absolute difference for the entire parameter space. We define **$\boldsymbol{\Delta}$-function** (delta-function) as:*

$$\Delta_k = \int \left(\mathcal{L}_k(\mathbf{w}) - \mathcal{L}_{k-1}(\mathbf{w})\right)^2 p(\mathbf{w}) dw,$$

where $p(\mathbf{w})$ describes the priority of the particular parameter points so we can make $p(\mathbf{w})$ have higher values next to the local minima.

We firther investigate this difference and aimed at exploration of how adding a new object to the dataset changes the value. We interested in convergence of this value and properties of loss function when the training dataset size limits to ∞.

Definition 4. *Let Δ be a positive hyperparameter that indicates the stop-difference for Δ_k. If*

$$k^* = \inf_k \{\forall m \geq k : \Delta_m < \Delta\}$$

*we can say that k^* samples in the dataset are enough to describe the distribution of data from the general population. We call k^* as **sufficient**.*

3.2 Assumptions

Assumption 1. *Let $\mathbf{w}^*$ be the local minimum of both $\mathcal{L}_{k-1}(\mathbf{w})$ and $\mathcal{L}_k(\mathbf{w})$. Thus,*

$$\nabla\mathcal{L}_{k-1}(\mathbf{w}^*) = \nabla\mathcal{L}_k(\mathbf{w}^*) = 0.$$

This assumption allows us to explore the behavior and the geometry of the loss function landscape at only one point.

Furthermore, using second-order Taylor's approximation for $\mathcal{L}_k(\mathbf{w})$ at $\mathbf{w}^*$ we get:

$$\mathcal{L}_k(\mathbf{w}) \approx \mathcal{L}_k(\mathbf{w}^*) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^*)^\mathsf{T}\mathbf{H}_k(\mathbf{w}^*)(\mathbf{w} - \mathbf{w}^*)$$

Assumption 2. *We can assume parameters $\mathbf{w}$ to be random, which will lead to quite natural condition: $p(\mathbf{w})$ can be even a prior distribution of $\mathbf{w}$, so:*

$$\Delta_k = \mathbb{E}_{p(\mathbf{w})}\left(\mathcal{L}_k(\mathbf{w}) - \mathcal{L}_{k-1}(\mathbf{w})\right)^2 = \mathbb{D}\left(\mathcal{L}_k(\mathbf{w}) - \mathcal{L}_{k-1}(\mathbf{w})\right) + \left(\mathbb{E}\left(\mathcal{L}_k(\mathbf{w}) - \mathcal{L}_{k-1}(\mathbf{w})\right)\right)^2$$

4 Method

As the work with the full Hessian is computation complex, let's define a projection of neural network parameters space Ω to a certain subspace Θ. Let $\mathbf{P}$ be a projection matrix from Ω to Θ. Let $\boldsymbol{\theta}$ be a projection: $\boldsymbol{\theta} = \mathbf{P}\mathbf{w}$.

Let $\mathbf{V} = \{\mathbf{e}_1, \dots, \mathbf{e}_d\}$ be eigenvectors of $\mathbf{H}$. We select d of them and project parameters space into a subspace on these d vectors. These vectors should represent maximum eigenvalues of $\mathbf{H}$ as these directions have the most impact to the loss function difference.

5 Experiments

To verify the theoretical estimates obtained, we conducted a detailed empirical study. In this section, we present experiments aimed at analyzing how the loss landscape evolves as the size of the training set increases.

Data. We use the MNIST dataset, which consists of 60,000 training images and 10,000 test images of handwritten digits (0–9). Each image is 28×28 pixels, grayscale.

Model Architecture. We use the following neural network models and preparations:

- A multilayer perceptron (MLP) with different number of hidden layers (ReLU activations).
- The choice of architecture remains consistent across all experiments.

The first two are preliminary experiments, conducted as illustrative examples on the MNIST dataset, while the others focusing on Hessian-based projections and show connections between eigenvalues and loss function convergence. That is our primary aim — to check our theoretical estimates in-deal.

5.1 Visualizing the Loss Landscape in a Random Two-Dimensional Subspace

The idea of this initial experiment is to get an intuitive sense of how the loss function's landscape shifts with different training-set sizes. To do this, we pick two random directions $\mathbf{v_1}$ and $\mathbf{v_2}$ in the parameter space and project the network's parameters onto the subspace they span.

Procedure

1. **Training on different sizes.** Train a neural network on increasingly larger subsets.
2. **Parameter snapshot.** After training, fix the final (or near-final) weights $\mathbf{w}^*$.
3. **Subspace grid.** Form a grid of points $\{\mathbf{w}^* + \alpha\mathbf{v}_1 + \beta\mathbf{v}_2\}$ around $\mathbf{w}^*$, where α and β vary in a chosen range.
4. **Loss evaluation.** Compute and plot the loss at each grid point, producing a 3D visualization.

For smaller N, the landscape exhibits sharper fluctuations. As N grows, the landscape often appears smoother and more stable.

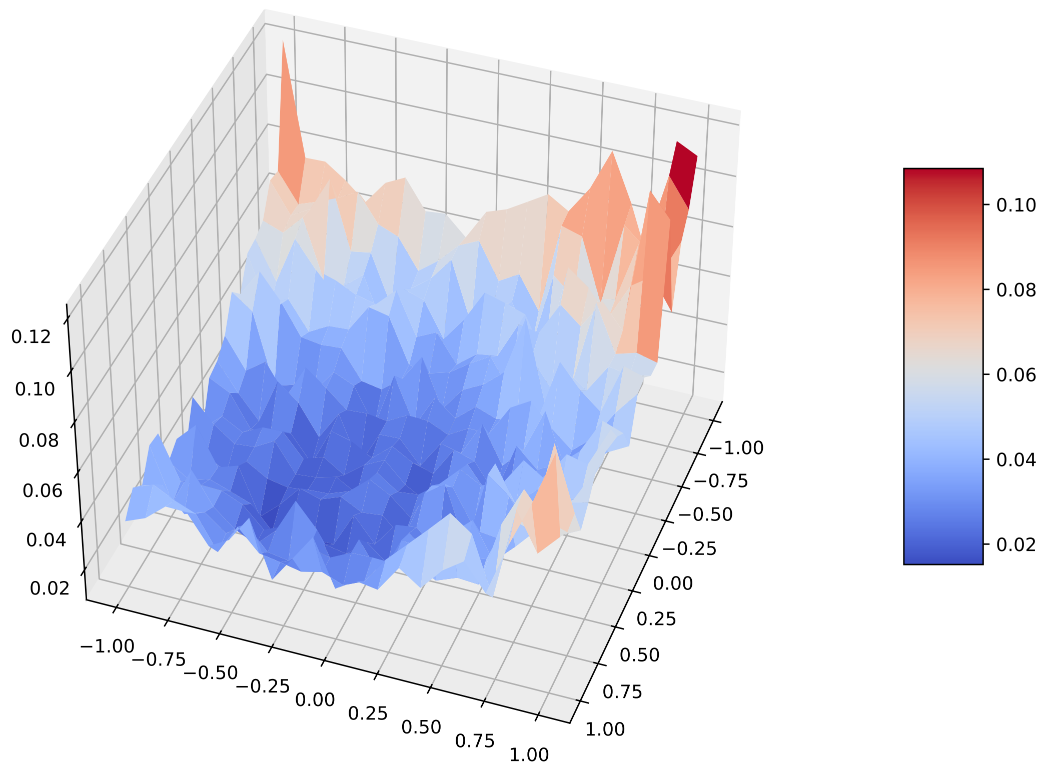

Figure 1: Loss function landscape

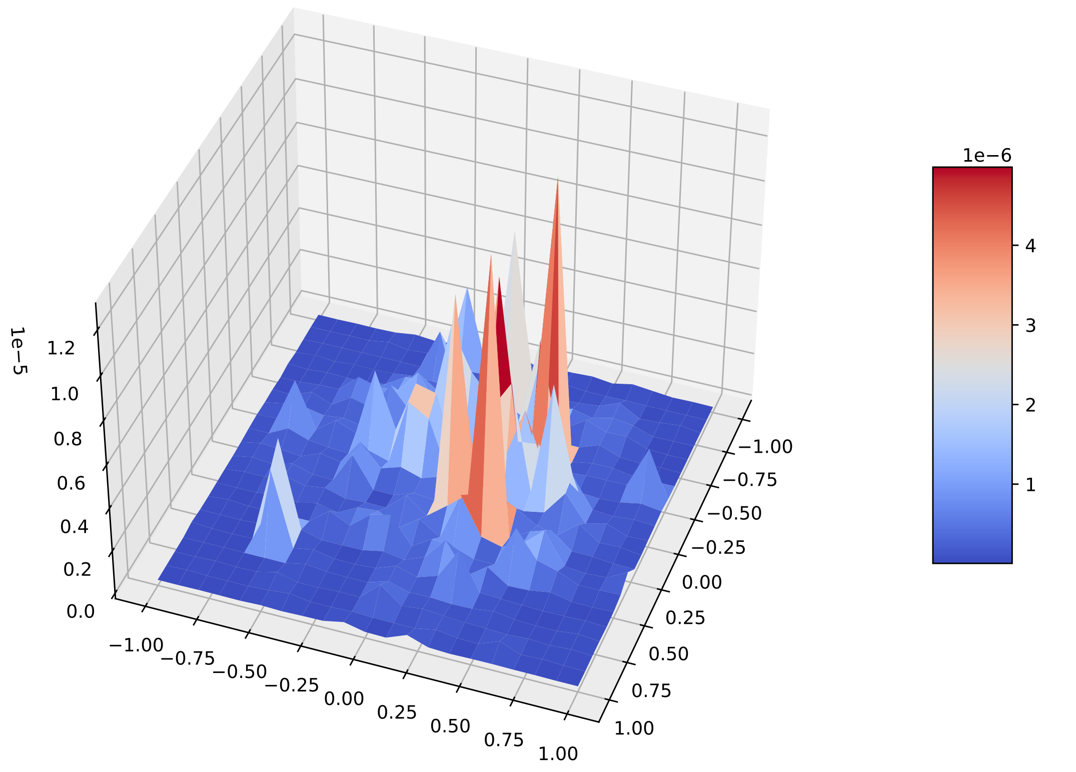

Figure 2: Loss function difference $(\mathcal{L}_k - \mathcal{L}_{k-1})^2$ landscape

5.2 Visualizing delta-function

Now we dig deeper and visualize our Δ_k function to clearly understand the difference in the loss function when moving from a model trained on $k-1$ samples to one trained on k samples. To do this we sample a bunch of random gaussian vectors (all components are normally distributed) and calculate a math expectation of modulus of the loss function difference. Directly computing this expectation over the entire distribution of random directions $\mathbf{d}$ is impossible, so we employ a Monte Carlo approximation: using Monte-Carlo method (we take the mean value of loss function on sampled gaussian shifted points):

$$\Delta_k = \mathbb{E}_{p(\mathbf{w})} \left(\mathcal{L}_k(\mathbf{w}) - \mathcal{L}_{k-1}(\mathbf{w})\right)^2 \approx \frac{1}{k} \sum_{t=1}^{k} \left(\mathcal{L}_k(\mathbf{w}^* + \mathbf{d}_t) - \mathcal{L}_{k-1}(\mathbf{w}^* + \mathbf{d}_t)\right)^2$$

where $\mathbf{d}_t \sim \mathcal{N}(0, I_d)$ are K random gaussian samples drawn from a multivariate normal distribution with zero mean and identity covariance from a subspace of rank d. Specifically, we proceed as follows:

1. **Sampling Directions.** Generate K independent directions $\mathbf{d}_1, \ldots, \mathbf{d}_M$ from $\mathcal{N}(0, I_d)$.
2. **Decreasing multidimension.** Project the network's parameters onto the subspace they span.
3. **Loss Evaluation.** For each direction $\mathbf{d}_t$, evaluate the loss at $\mathbf{w}^* + \mathbf{d}_t$ and
$$\left(\mathcal{L}_k(\mathbf{w} + \mathbf{d}_t) - \mathcal{L}_{k-1}(\mathbf{w} + \mathbf{d}_t)\right)^2$$
4. **Averaging.** Average these values values over $k = 1, \ldots, K$ to approximate Δ_k.

By increasing K, we obtain a more reliable estimate of the expected difference in loss landscapes. This Monte Carlo approach is straightforward to implement and does not require explicit knowledge of the underlying loss function's distribution.

The Δ-function generally decreases as N increases, suggesting that the landscape differences between neighbouring sizes become smaller. Beyond a certain point, Δ-values are nearly negligible, indicating that adding more data has minimal impact on the local landscape in these directions.

Conclusion from Preliminary Experiments. These initial experiments demonstrate that the loss landscape becomes progressively more stable with increasing N. However, the directions used here are either random. This motivates a more rigorous approach in the *main experiment* below, where we focus specifically on the most critical directions of curvature — those corresponding to the top eigenvalues of the Hessian.

5.3 Main Experiment: Projection onto Dominant Hessian Eigenvectors

From prior work [7], we know that only a few Hessian eigenvectors (those associated with the largest eigenvalues) tend to capture the most significant curvature directions. By projecting onto these principal directions, we can more precisely measure how the landscape the loss surface changes as k increases.

Procedure

1. **Hessian approximation.** For each trained model, approximate the Hessian $\mathbf{H}$ or its dominant eigenvalues/eigenvectors. Techniques ???
2. **Eigen-decomposition.** Identify the top d eigenvectors $\{\mathbf{e}_1, \ldots, \mathbf{e}_d\}$ with the largest eigenvalues.
3. **Calculation and analization.** For each model, calculate a Δ fuction via the same way as in the second experiment (using Monte-Carlo) and visualize via plots.

Goal and Expected Outcome

- By focusing on the directions of largest curvature, we expect a clearer measure of how the landscape of the loss surface changes with additional data.

- This approach should yield a more accurate estimate of the point at which the landscape effectively stabilizes, thus helping to pinpoint a minimum viable dataset size.
- By applying experiments we check our calculations above on the reality.

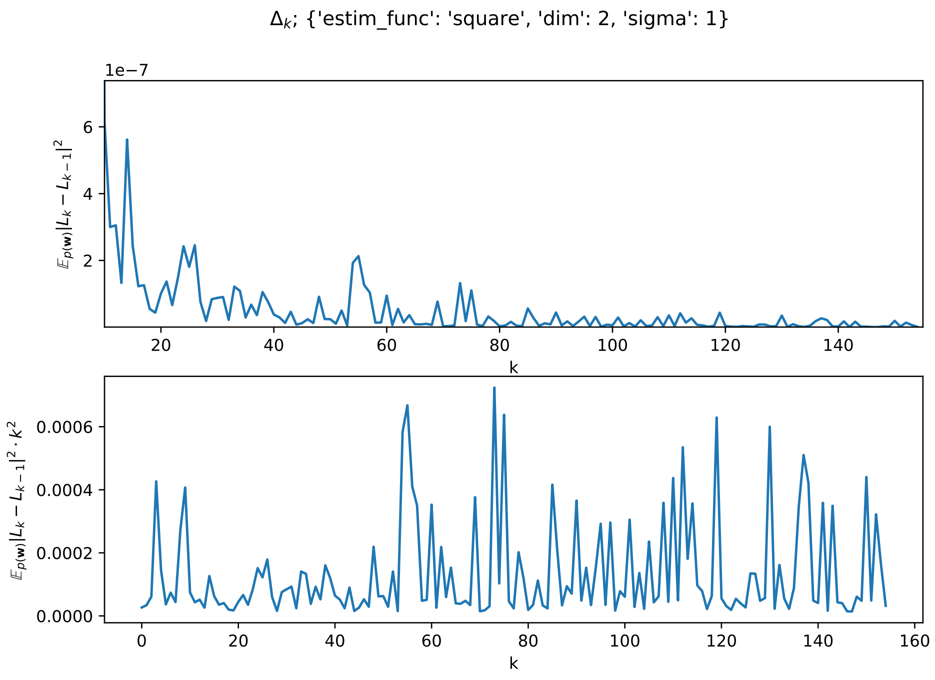

Figure 3: Δ-function visualization. ‘Sigma‘ parameter refers to the variance of sampled gaussian vectors, ‘dim’ refers to the rank of the subspace.

6 Discussion

TADAAA!

7 Conclusion

TADAAA!

References

[1] Derya Soydaner. A comparison of optimization algorithms for deep learning. *International Journal of Pattern Recognition and Artificial Intelligence*, 34(13):2052013, April 2020. ISSN 1793-6381. doi: 10.1142/s0218001420520138. URL `http://dx.doi.org/10.1142/S0218001420520138`.

[2] Lei Wu, Zhanxing Zhu, and Weinan E. Towards understanding generalization of deep learning: Perspective of loss landscapes. *preprint arXiv:1706.10239*, 2017. Characteristics of the loss landscape explain the good generalization capability.

[3] Kaare Brandt Petersen and Michael Syskind Pedersen. The matrix cookbook. `https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf`, 2012. A collection of facts about matrices and matters relating to them.

[4] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6), 2012. URL `https://api.semanticscholar.org/CorpusID:5280072`. The database of handwritten digits.

[5] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *preprint arXiv:1708.07747*, 2017. A dataset of Zalando's article images.

[6] Nikita Kiselev and Andrey Grabovoy. Unraveling the hessian: A key to smooth convergence in loss function landscapes. *arXiv preprint arXiv:2409.11995*, 2024. Upper bounds via Hessian for fully connected neural networks.

[7] Levent Sagun, Utku Evci, Ugur Guney, Yann Dauphin, and Leon Bottou. Empirical analysis of the hessian of over-parametrized neural networks. *preprint arXiv:1706.04454*, 2018. In over-trained neural networks the Hessian rank is often low; the active subspace may be significantly smaller.

[8] Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. In *International Conference on Machine Learning*, 2019. URL `https://proceedings.mlr.press/v97/ghorbani19b/ghorbani19b.pdf`. Observed large isolated eigenvalues in the spectrum and a surprising concentration of the gradient in the corresponding eigenspaces.

[9] Vladislav Meshkov, Nikita Kiselev, and Andrey Grabovoy. Convnets landscape convergence: Hessian-based analysis of matricized networks, 2024. URL `https://ieeexplore.ieee.org/document/10899113`. Upper bounds via Hessian for convolutional neural networks.

[10] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *preprint arXiv:2203.15556*, 2022. Motivations behind overloaded models and the importance of optimal training size estimation.

[11] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *preprint arXiv:1712.09913*, 2018. Methods of visualizing loss function curvature and comparing different functions.

A Appendix / supplemental material

A.1 Additional experiments / Proofs of Theorems

TADAAA!