Evaluating Loss Landscapes from a Topology Perspective

Tiankai Xie[*]
Arizona State University
`txie21@asu.edu`

Caleb Geniesse[*]
Lawrence Berkeley National Lab
`cgeniesse@lbl.gov`

Jiaqing Chen[*]
Arizona State University
`jchen501@asu.edu`

Yaoqing Yang
Dartmouth College
`yaoqing.yang@dartmouth.edu`

Dmitriy Morozov
Lawrence Berkeley National Lab
`dmorozov@lbl.gov`

Michael W. Mahoney
ICSI, LBNL, and UC Berkeley
`mmahoney@stat.berkeley.edu`

Ross Maciejewski
Arizona State University
`rmacieje@asu.edu`

Gunther H. Weber
Lawrence Berkeley National Lab
`ghweber@lbl.gov`

Abstract

Characterizing the loss of a neural network with respect to model parameters, i.e., the *loss landscape*, can provide valuable insights into properties of that model. Various methods for visualizing loss landscapes have been proposed, but less emphasis has been placed on quantifying and extracting actionable and reproducible insights from these complex representations. Inspired by powerful tools from topological data analysis (TDA) for summarizing the structure of high-dimensional data, here we characterize the underlying *shape* (or topology) of loss landscapes, quantifying the topology to reveal new insights about neural networks. To relate our findings to the machine learning (ML) literature, we compute simple performance metrics (e.g., accuracy, error), and we characterize the local structure of loss landscapes using Hessian-based metrics (e.g., largest eigenvalue, trace, eigenvalue spectral density). Following this approach, we study established models from image pattern recognition (e.g., ResNets) and scientific ML (e.g., physics-informed neural networks), and we show how quantifying the shape of loss landscapes can provide new insights into model performance and learning dynamics.

1 Introduction

Given the important role that the loss function plays during learning, examining it with respect to a neural network's weights—by visualizing the so-called *loss landscape*—can provide valuable insights into both network architecture and machine learning (ML) dynamics [Martin and Mahoney, 2021, Martin et al., 2021, Yang et al., 2022b, 2021, Zhou et al., 2023]. Indeed, the loss landscape has been essential for understanding certain aspects of deep learning, including, but not limited to, test accuracy, robustness of transfer learning [Djolonga et al., 2021], robustness to out-of-distribution

[*]Equal contribution.

detection [Yang et al., 2022a], robustness to adversarial attack [Kurakin et al., 2016], and generalizability [Cha et al., 2021]. There are two popular approaches to generating the loss landscape for a given neural network model. Initial efforts to visualize the loss landscape relied on sampling random orthogonal vectors and projecting weights onto the plane spanned by these random vectors [Goodfellow et al., 2014, Li et al., 2018]. More recently, Yao et al. [2020] proposed using directions based on the Hessian, wherein the first two most important Hessian eigenvectors are used to capture more meaningful changes in the loss function. In both approaches, a neural network's parameters are perturbed along each direction, and the loss is re-evaluated at each of these positions.

While both approaches have provided valuable insights, *loss landscape visualization* (no matter which method was used) is often limited to just that—visualization. In other words, loss landscapes, once created, are often simply visually explored or qualitatively compared. It is less clear how to meaningfully measure or quantitatively relate these landscapes to features of the model's underlying architecture or to properties inherent to the learning process. Indeed, examining and quantifying a loss landscape—which is inherently high-dimensional, with as many dimensions as the number of parameters in the model—is challenging to do, especially when using two-dimensional views and qualitative observations alone.

To provide a more quantitative approach to understanding and using loss landscapes, here we show how topological data analysis (TDA) can be used to quantify and extract (quantitative) insights based on the topology (or shape) of those landscapes. We first compute loss landscapes using either random projections or Hessian-based directions and explore four different representations, including one image data representation (where the loss is stored as pixels) and three unstructured grid representations (where the loss is stored on the vertices of a graph). We then apply two methods from TDA, namely, the merge tree [Carr et al., 2003, Heine et al., 2016] and persistence diagram [Edelsbrunner and Harer, 2008], to quantify and compare different loss landscapes. We quantify these structures by measuring the number of saddle points and average persistence, respectively, and we compare our results with state-of-the-art methods for evaluating model performance, as well as with more recent methods for evaluating the local geometry of loss landscapes based on the Hessian.

2 Background on TDA

Topological data analysis (TDA) aims to reveal the global underlying structure of data. It is particularly useful for studying high-dimensional data or functions, where direct visualization is impossible. Here, we leverage ideas and algorithms from TDA to study the structure of the loss function, i.e., the so-called loss landscape. In the context of a loss function, we are interested in the number of minima (i.e., unique sets of parameters for which the loss is locally minimized) and how "prominent" they are (i.e., measuring how many other sets of neighboring parameters have a higher loss than the parameter sets that locally minimize the loss function). Such information can be obtained from the merge tree and persistence diagram (i.e., captured by the 0-dimensional persistent homology).

A *merge tree* [Carr et al., 2003, Heine et al., 2016] tracks connected components of sub-level sets $L^{-}(v) = \{x \in D; x \leq v\}$ as a threshold, v, is increased. Note, the merge tree can track either sub-level or super-level sets, but here we are interested in characterizing loss functions and their minima, so we focus on sub-level sets. In this case, as v increases, new connected components form at local minima and later merge with neighboring connected components (other local minima) at saddles. The merge tree encodes these changes in the loss landscape as nodes in a tree-like structure, where local minima are represented by degree-one nodes (connected to other local minima through a single saddle point), and the saddle points connecting different minima are represented by degree-three nodes (each connecting two local minima and one other saddle point).

A *persistence diagram* represents features (i.e., branches in the merge tree) as points in a two-dimensional plane. The horizontal axis corresponds to the birth of each feature—which is the value of the minimum at which it first appears. The vertical axis corresponds to the death of each feature—which is the value of the saddle where it merges into a more persistent feature. The distance between a point and the diagonal line $y = x$ encodes the persistence of the feature—which is a measure of how long the feature lasts in the filtration (i.e., as the threshold, v, is increased). Such a metric captures information about the ruggedness of the landscape, and for example, the depth of local valleys and the height of the barriers between them. Here, we quantify the average persistence by computing the distance between each persistence pair and the diagonal and then taking the average.

3 Empirical Results

In this section, we provide a summary of our main empirical results. We study established models from image pattern recognition (e.g., ResNets) and scientific ML (e.g., physics-informed neural networks). We first show how removing residual connections from ResNet changes the shape of the loss landscape. We then show how the loss landscape changes for a physics-informed neural network as a physical parameter is varied and optimization begins to fail. In both experiments, we quantify the merge tree and persistence diagrams, and we relate our results to model performance and loss landscape metrics like the top Hessian eigenvalue and the Hessian trace.

3.1 Image Pattern Recognition

In our first experiment, we explore image pattern recognition using ResNet-20 trained on CIFAR-10 [Yao et al., 2020]. Specifically, we look at (and quantify) loss landscapes before and after adding residual connections. Recent work shows that the residual connections are related to the "smoothness" of the loss landscape [Li et al., 2018, Yao et al., 2020]. Here, we aim to verify this and further characterize how the residual connections in ResNet-20 change the underlying loss landscape. Note, we removed the residual connections from ResNet-20 before training. The accuracy of ResNet-20 without residual connections (90% average accuracy across four random seeds) was slightly lower than ResNet-20 with residual connections (92% average accuracy across four random seeds).

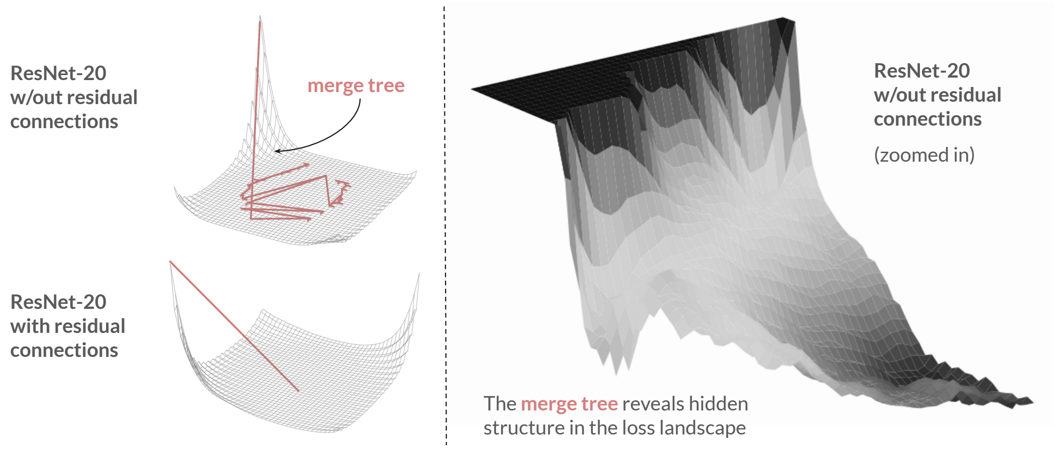

Figure 1: Visualizing loss landscapes for ResNet-20. Compare with Yao et al. [2020].

In Fig. 1, we show loss landscapes for ResNet without and with residual connections. We observe a much more complicated structure in the loss landscape for ResNet-20 without residual connections, revealed by the more complicated branched structure of the merge tree. Interestingly, this complicated structure is not immediately visible in the loss landscape itself. On the right, we show the same landscape for ResNet-20 without residual connections, after clipping outlier values to "zoom" into the smaller scale structure. The misleading visualization on the left highlights a unique benefit of our approach. That is, the merge tree can capture interesting structure across different scales by default, requiring less manual fine-tuning of visualization parameters. Comparatively, we observe a much smoother loss landscape for ResNet-20 with the residual connections, confirmed by the simpler structure of the merge tree (i.e., a single minima and no saddle points). Interestingly, these observations agree with previous findings that adding residual connections to ResNet results in a "smoother" loss landscape, thereby improving generalization [Li et al., 2018].

In Fig. 5 (in the appendix), we further verify these observations numerically by showing how our TDA-based metrics relate to ML-based metrics. These plots provide additional insights beyond the qualitative differences in the loss landscapes we observed, revealing that the number of saddle points in the merge tree increases and the average persistence decreases when the residual connections are removed from ResNet-20. Looking across the different columns, we observe an inverse relationship between the number of saddle points in the merge tree and the ML-based metrics, but a direct relationship between the average persistence and the same ML-based metrics. Together, these results provide additional insight into how changing the architecture of a neural network like ResNet-20 can result in a “smoother” (and thereby easier to optimize) loss landscape.

3.2 Physics-Informed Neural Networks

In our second experiment, we look at a set of physics-informed neural network (PINN) models trained to solve increasingly difficult convection problems. Specifically, we consider the one-dimensional convection problem, a hyperbolic partial differential equation that is commonly used to model transport phenomena:

$$\frac{\partial u}{\partial t} + \beta \frac{\partial u}{\partial x} = 0, \quad x \in \Omega, \quad t \in [0, T] \tag{1}$$

$$u(x, 0) = h(x), \quad x \in \Omega, \tag{2}$$

where β is the convection coefficient and $h(x)$ is the initial condition. The general loss function for this problem is

$$L(\theta) = \frac{1}{N_u} \sum_{i=1}^{N_u} (\hat{u} - u_0^i)^2 + \frac{1}{N_f} \sum_{i=1}^{N_f} \lambda_i (\frac{\partial \hat{u}}{\partial t} + \beta \frac{\partial \hat{u}}{\partial x})^2 + L_B, \tag{3}$$

where $\hat{u} = NN(\theta, x, t)$ is the output of the NN, and L_B is the boundary loss. The goal of this case study is to investigate the PINN's soft regularization and how it helps (or fails to help) the optimizer find optimal solutions to seemingly simple convection problems. As shown in Krishnapriyan et al. [2021], increasing the wave speed parameter, β, can make it harder for the PINN to find a reasonable solution. This difficulty has been linked to changes in the loss landscape, which becomes increasingly complicated, such that optimizing the model becomes increasingly difficult.

In Fig. 2, we show that increasing the wave speed indeed results in a more complicated loss landscape. In the bottom row, we show the spatiotemporal patterns predicted by the PINN models. Note, increasing β makes the problem harder to solve, and here we can see that our PINN models fail to

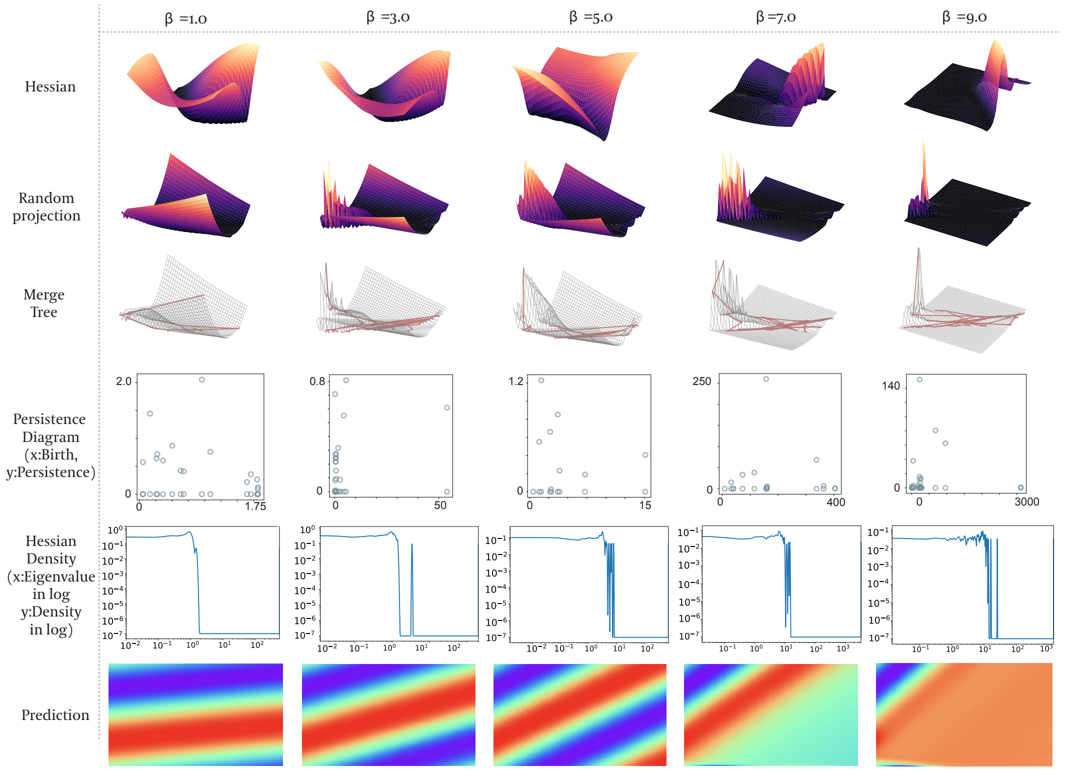

Figure 2: Visualizing the failure modes of PINNs. Compare with Krishnapriyan et al. [2021].

accurately predict the pattern around $\beta = 7.0$. Looking at the first three rows, we observe increasingly complex loss landscapes as β increases, in both versions of the loss landscape and further revealed by the more complicated and branched structure of the merge tree (i.e., with many more minima and saddle points). The change in complexity of the landscape is further supported by higher persistence values in the persistence diagrams. We also relate our observations to the Hessian Density, which suggests that the volume and complexity of the loss landscape increases with β. Together, these different views give us a more holistic understanding of how the loss landscape changes as β increases. Interestingly, these observations agree with previous findings that increasing β results in a more complicated loss landscape, corresponding to a harder optimization problem [Krishnapriyan et al., 2021]. We further verify these observations numerically in Fig. 6 (in the appendix). These plots provide additional insights beyond the qualitative changes in the loss landscapes we observed, confirming that the number of saddle points in the merge tree (left column) increases, along with the average persistence (right column), as the value of β increases. We observe similar trends in the Absolute Error, Top-1 Hessian Eigenvalue, and Hessian Trace as β increases. Together, these results reaffirm previous findings and provide new insights into the failure modes of PINNs, revealing that the topology of the loss landscape becomes significantly complex as PINNs start to fail.

4 Conclusion and Future Work

In this work, we study neural network loss landscapes through the lens of TDA. Specifically, we capture the underlying shape of loss landscapes using merge trees and persistence diagrams. By quantifying these topological constructs, we reveal new insights about the landscapes. Furthermore, we explore the relationship between our TDA-based metrics and relevant traditional ML metrics. For ResNet models, we find that the number of saddle points is inversely related to the average persistence and to the other ML-based metrics. For PINN models, we find that the number of saddle points and average persistence increase together along with the other ML-based metrics. These relationships reflect the curvature and sharpness of the landscape, which in turn strongly impacts the model's performance and generalization abilities. Note, in this work we only show the 0-dimensional persistence diagram, which is exactly what the branches in the merge tree encode. Since here our original focus was on extracting merge trees, we decided to limit our analysis to these lower-dimensional features. We leave the analysis of higher-dimensional holes for future work.

Moreover, since the merge tree and persistence diagram can be computed for arbitrary-dimensional spaces, our generalized and scalable approach opens up the door to studying loss landscapes in higher dimensions. In the future, we hope to extend our approach to characterize higher-dimensional loss landscapes, under the hypothesis that more information (hidden in these additional dimensions) could perhaps provide new insights into the loss function and properties relating neural network architecture to learning dynamics. One simple way to do this would be to sample along more directions. So, for example, we could construct a subspace based on the top 3 to 10 Hessian eigenvectors. While this would still be far from the potentially billions of dimensions in the true high-dimensional loss landscapes of modern ML models (with potentially billions of parameters), we expect that there exists a much lower-dimensional manifold upon which the interesting variation can be observed.

5 Acknowledgments

This work was supported by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research (ASCR) program under Contract Number DE-AC02-05CH11231 to Lawrence Berkeley National Laboratory and Award Number DE-SC0023328 to Arizona State University ("Visualizing High-dimensional Functions in Scientific Machine Learning"). This research used resources at the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy, Office of Science, User Facility under NERSC Award Number ASCR-ERCAP0026937.

References

Talha Bin Masood, Joseph Budin, Martin Falk, Guillaume Favelier, Christoph Garth, Charles Gueunet, Pierre Guillou, Lutz Hofmann, Petar Hristov, Adhitya Kamakshidasan, Christopher Kappe, Pavol Klacansky, Patrick Laurin, Joshua Levine, Jonas Lukasczyk, Daisuke Sakurai, Maxime Soler, Peter Steneteg, Julien Tierny, Will Usher, Jules Vidal, and Michal Wozniak. An overview of the

Topology ToolKit. In *Topological Methods in Data Analysis and Visualization VI*, pages 327–342. Springer International Publishing, 2021.

Hamish Carr, Jack Snoeyink, and Ulrike Axen. Computing contour trees in all dimensions. *Computational Geometry*, 24(2):75–94, 2003. ISSN 0925-7721. Special Issue on the Fourth CGC Workshop on Computational Geometry.

Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. Swad: Domain generalization by seeking flat minima. *Advances in Neural Information Processing Systems*, 34:22405–22418, 2021.

Josip Djolonga, Jessica Yung, Michael Tschannen, Rob Romijnders, Lucas Beyer, Alexander Kolesnikov, Joan Puigcerver, Matthias Minderer, Alexander D'Amour, Dan Moldovan, et al. On robustness and transferability of convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16458–16468, 2021.

Wei Dong, Charikar Moses, and Kai Li. Efficient k-nearest neighbor graph construction for generic similarity measures. In *Proceedings of the 20th international conference on World wide web*, pages 577–586, 2011.

Herbert Edelsbrunner and John Harer. Persistent Homology-a Survey. *Contemporary mathematics*, 453(26):257–282, 2008.

Ian J Goodfellow, Oriol Vinyals, and Andrew M Saxe. Qualitatively characterizing neural network optimization problems. *arXiv preprint arXiv:1412.6544*, 2014.

Christian Heine, Heike Leitte, Mario Hlawitschka, Federico Iuricich, Leila De Floriani, Gerik Scheuermann, Hans Hagen, and Christoph Garth. A Survey of Topology-based Methods in Visualization. *Computer Graphics Forum*, 35(3):643–667, 2016.

Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, 34:26548–26560, 2021.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 2018.

Charles H Martin and Michael W Mahoney. Implicit Self-Regularization in Deep Neural Networks: Evidence from Random Matrix Theory and Implications for Learning. *The Journal of Machine Learning Research*, 22(1):7479–7551, 2021.

Charles H Martin, Tongsu Peng, and Michael W Mahoney. Predicting trends in the quality of state-of-the-art neural networks without access to training or testing data. *Nature Communications*, 12(1):4122, 2021.

Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized out-of-distribution detection: A survey, 2022a.

Yaoqing Yang, Liam Hodgkinson, Ryan Theisen, Joe Zou, Joseph E Gonzalez, Kannan Ramchandran, and Michael W Mahoney. Taxonomizing local versus global structure in neural network loss landscapes. *Advances in Neural Information Processing Systems*, pages 18722–18733, 2021.

Yaoqing Yang, Ryan Theisen, Liam Hodgkinson, Joseph E Gonzalez, Kannan Ramchandran, Charles H Martin, and Michael W Mahoney. Evaluating natural language processing models with generalization metrics that do not need access to any training or testing data. *arXiv preprint arXiv:2202.02842*, 2022b.

Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael W Mahoney. PyHessian: Neural networks through the lens of the hessian. In *2020 IEEE international conference on big data (Big data)*, pages 581–590. IEEE, 2020.

Yefan Zhou, Yaoqing Yang, Arin Chang, and Michael W Mahoney. A three-regime model of network pruning. In *International Conference on Machine Learning*, pages 42790–42809. PMLR, 2023.

A Appendix

A.1 Loss Landscapes

To compute a loss landscape, we need to project the high-dimensional loss function into a two-dimensional space. To construct this space, we therefore need to sample two orthonormal vectors. We consider two different ways to sample these vectors (e.g., random directions, Hessian-based directions), but the rest of the procedure for sampling the loss function in the resulting spaces is the same regardless of how the directions were chosen.

The first (naive) way to define the lower-dimensional subspace is to sample two random vectors. In high dimensions, they will most likely be nearly orthogonal. Here, we use PyTorch to randomly sample two vectors having the same size as the number of parameters in the model.

The second way to define the lower-dimensional subspace is to use the top two Hessian eigenvectors. Since the eigenvectors associated with the largest eigenvalues correspond to the directions of most variation in the loss function, sampling along these directions can often reveal more interesting surfaces. Here, we use PyHessian [Yao et al., 2020], a PyTorch library for efficiently calculating Hessian information using (randomized) numerical linear algebra.

Given the two orthonormal directions, we follow the approach outlined by Li et al. [2018]. More formally, we sample the trained model at discrete positions in the subspace defined by the two directions and evaluate the loss $\mathscr{L}$ as follows:

$$f(\alpha_1, \alpha_2) = \mathscr{L}(\theta + \alpha_1 \delta_1 + \alpha_2 \delta_2), \tag{4}$$

where (α_1, α_2) are the coordinates in the two-dimensional subspace, δ_1 and δ_2 correspond to the first and second direction defining that subspace, and θ is the original model. As such, each coordinate corresponds to a perturbed model, and we store the loss for that model, such that the collection of loss values forms the two-dimensional loss landscape.

Given a two-dimensional loss landscape, we can represent the sampled points as *image data*, where each pixel represents the loss of a perturbed model, or as an *unstructured grid*, where each vertex in the grid is associated with a scalar loss value. For the unstructured grid, we need to define the spatial proximity (or connectivity) of vertices in the grid based on the similarity of their coordinates. This will allow us to characterize how the loss changes throughout the landscape (i.e., as parameters are perturbed from one vertex to the next).

Here we explored both triangulations (e.g., Delaunay, Gabriel graph) and neighbor-based graph approaches. We found that a scalable, approximate nearest neighbor algorithm [Dong et al., 2011] yielded good results while also being much faster. Specifically, we use a k-nearest neighbor graph, where each point is connected to the k most similar points, and we use $k = 8$, such that the connectivity is similar to the spatial proximity of pixels in an image (i.e., left, right, top, bottom, and all four corners).

A.2 Topological Data Analysis

We perform topological data analysis (TDA) to extract and summarize the most important features of the loss landscapes. We then further quantify the results from TDA to provide new metrics that can be related to things like model performance. Specifically, we compute the merge tree and the 0-dimensional persistence diagram. Recall that the branches in the merge tree are equivalent to the 0-dimensional features [Edelsbrunner and Harer, 2008]. However, compared to the persistence diagram which describes the life span of each component, the merge tree provides additional information including which component each component merges into. The merge tree also provides a nice intuitive visualization of the changes in the loss landscape. To quantify the merge tree, we count the number of saddle points and the number of minima. To quantify the persistence diagram, we measure the average persistence by computing the distance between each persistence pair and the diagonal and then taking the average.

Note, we compute the merge tree and persistence diagram for each loss landscape using the Topology ToolKit (TTK) [Bin Masood et al., 2021]. Further analysis and visualization of the resulting merge trees and persistence diagrams was performed in Python.

B Supplemental Figures

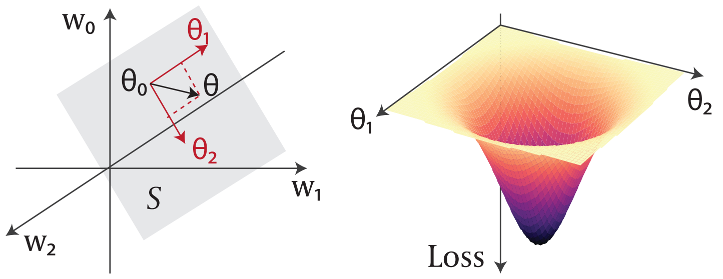

Figure 3: Illustration of generating the two-dimensional loss landscape. In this work, to compute a loss landscape, we first construct a subspace defined by two vectors, θ_1 and θ_2. Within this subspace, we perform model interpolation and evaluate the corresponding loss values.

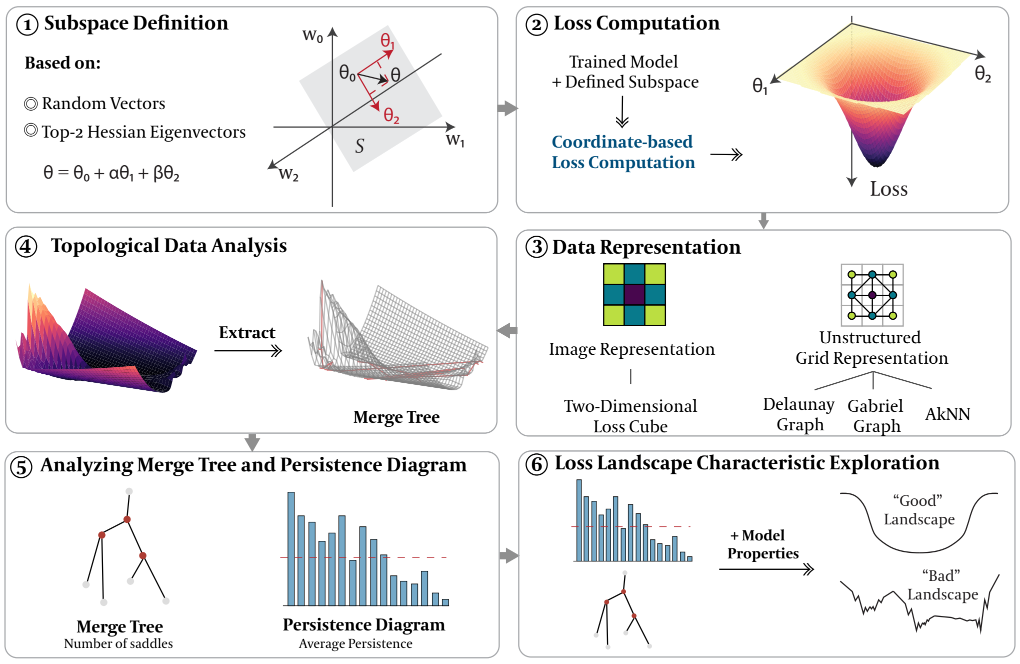

Figure 4: Our loss landscape analysis pipeline. The pipeline includes six stages: (1) *Subspace Definition*: Define the loss landscape subspace using random-based or Hessian-based directions. (2) *Loss Computation*: Calculate loss values for coordinate locations using our coordinate-based loss computation method. (3) *Data Representation*: Transform the loss landscape into suitable data structures for TDA. (4) *Topological Analysis*: Extract the merge tree and persistence diagram. (5) *Quantitative Evaluation*: Quantify the merge tree and persistence diagram. (6) *Loss Landscape Property Evaluation*: Relate the TDA-based metrics to loss landscape properties.

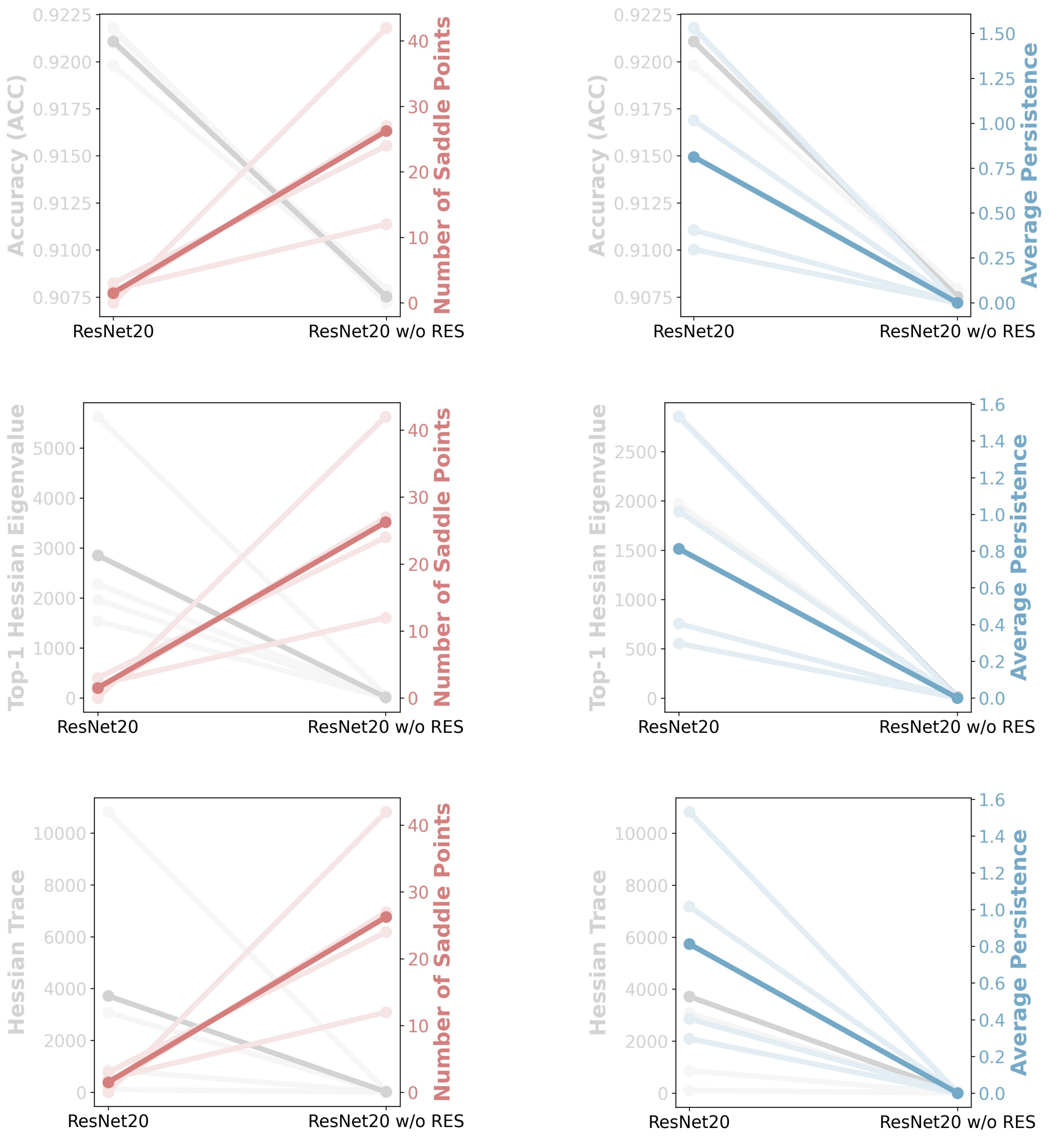

Figure 5: Quantifying the loss landscape for ResNet-20, with and without residual connections. We trained each version of the model four separate times, each time using a unique random seed (e.g., `0`, `123`, `123456`, and `2023`). Here, we numerically verify the observations we made in Fig. 1 and provide additional insights based on persistence diagrams (not shown). We quantified the merge tree and persistence diagram by counting the number of saddle points and computing the average persistence, respectively. We compare our results with traditional machine learning metrics, including the Accuracy, Top-1 Hessian Eigenvalue, and Hessian Trace. Here, we show the relationship between those ML-based metrics and our two TDA-based metrics. These plots provide additional insights beyond the qualitative differences in the loss landscapes we observed in Fig. 1, confirming that the landscapes for ResNet-20 models without residual connections correspond to merge trees with a higher number of saddle points (left column). In contrast, we see that these models (without residual connections) display a lower average persistence (right column). We observe an inverse relationship between the number of saddle points in the merge tree and the ML-based metrics, but a direct relationship between the average persistence and the same ML-based metrics. Together, these results provide insight into how changing the architecture of a neural network like ResNet-20 (i.e., by adding residual connections) can result in a “smoother” (and thereby easier to optimize) loss landscape.

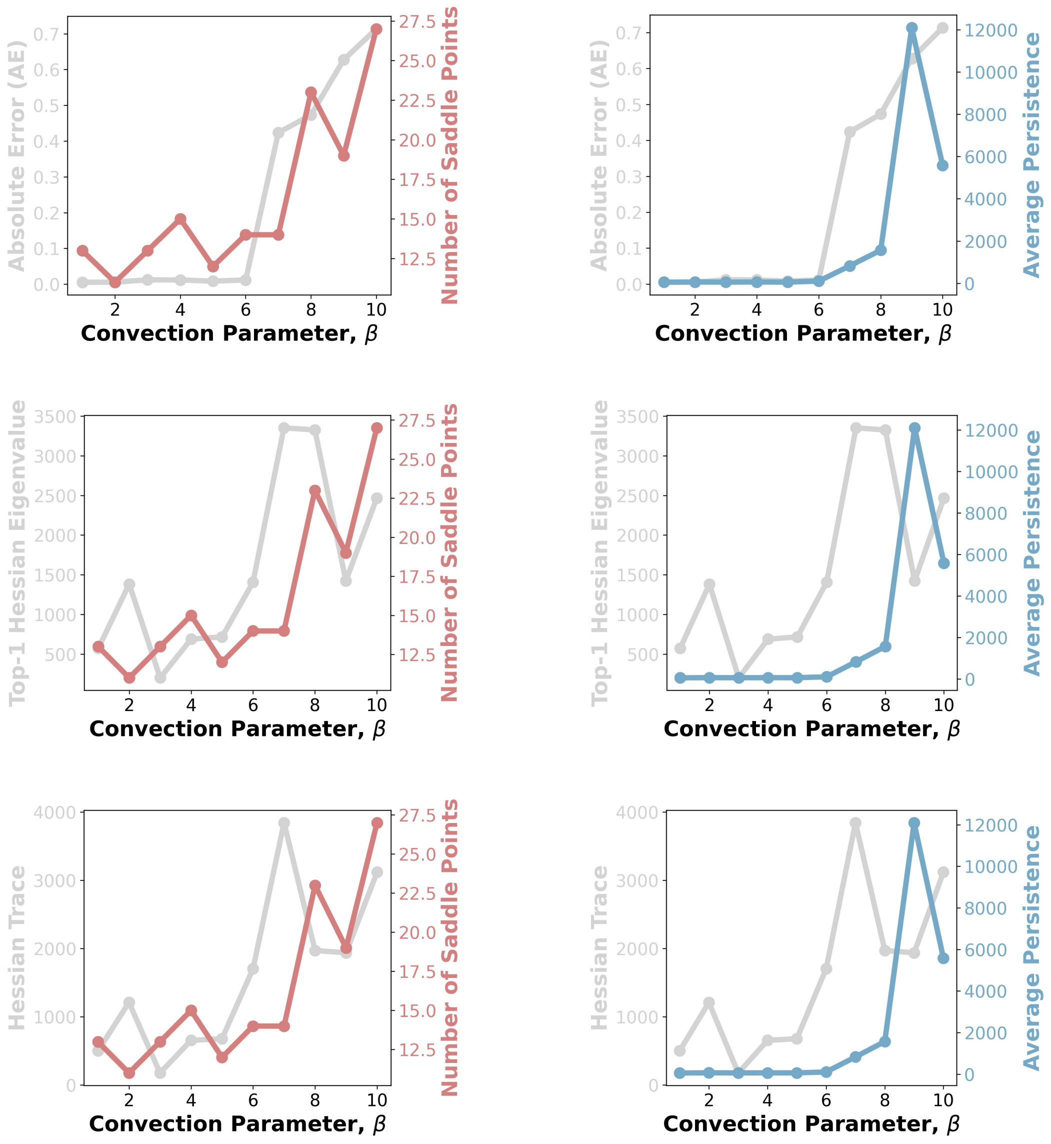

Figure 6: Quantifying the failure modes of PINNs. Here we show additional experimental results for PINN models solving increasingly difficult convection problems (Eq. 3 for $\beta \in [1..10]$) using a fixed random seed (*seed* $= 0$). We show our proposed TDA-based metrics, including the number of saddle points in the merge tree and average persistence, along with more traditional ML-based metrics, including the Absolute Error, Top-1 Hessian Eigenvalue, and Hessian Trace. These plots provide additional insights beyond the qualitative changes in the loss landscapes we observed in Fig. 2, confirming that the number of saddle points in the merge tree (left column) increases, along with the average persistence (right column), as the value of β increases. In each row, we overlay our TDA-based metrics with a different ML-based metric. We observe similar trends in the Absolute Error, Top-1 Hessian Eigenvalue, and Hessian Trace as β increases. Together, these results reaffirm previous findings and provide new insights into the failure modes of PINNs, suggesting that the topology of the loss landscape becomes significantly more complex and difficult to optimize.