Topological Analysis of Neural Network Loss landscapes

A PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

Master of Technology

IN

Computer Science and Engineering

BY

Pankaj Meena

Computer Science and Automation
Indian Institute of Science
Bangalore – 560 012 (INDIA)

June, 2024

Declaration of Originality

I, **Pankaj Meena**, with SR No. **04-04-00-10-51-22-1-21598** hereby declare that the material presented in the thesis titled

Topological Analysis of Neural Network Loss landscapes

represents original work carried out by me in the **Department of Computer Science and Automation** at **Indian Institute of Science** during the years **2023-24**.
With my signature, I certify that:

- I have not manipulated any of the data or results.
- I have not committed any plagiarism of intellectual property. I have clearly indicated and referenced the contributions of others.
- I have explicitly acknowledged all collaborative research and discussions.
- I have understood that any false claim will result in severe disciplinary action.
- I have understood that the work may be screened for any form of academic misconduct.

Date: 21/06/2024 Student Signature

In my capacity as supervisor of the above-mentioned work, I certify that the above statements are true to the best of my knowledge, and I have carried out due diligence to ensure the originality of the report.

Advisor Name: **Prof. Vijay Natarajan** Advisor Signature

DEDICATED TO

The Research Community

that continually strives for new discoveries to answer previously unanswered questions

Acknowledgements

Grateful acknowledgment is extended to Prof. Vijay Natarajan and Dr. Harish Doraiswamy for their valuable guidance and constructive feedback throughout the project. We have used the chatGPT tool for improving the language and style of the document. We have also used it to summarise a few documents but we have verified the veracity of the summaries provided by chatGPT.

Abstract

A neural network's capacity to generalize effectively depends on crucial factors such as its initial learning rate, the regularization provided by weight decay, the choice of batch size for training, and the strategic timing of adjustments to the learning rate throughout the training phase.

But how exactly do these factors affect how well the model can adapt to new situations? To find out, we need to dive into the inner workings of neural networks. In this project, we'll take a deep dive into the network's "landscape" of loss in high dimension, examining how it changes over time. By carefully examining varying initial learning rates cases on WideResnet-16, we aim to decipher the factors influencing differing levels of generalization. By scrutinizing how these factors affect the model's ability to adapt to new data, we seek insights into optimizing its performance across various tasks.

Contents

List of Figures

LIST OF FIGURES

Chapter 1

Introduction

Loss Landscapes play a crucial role in training neural networks. Several papers argue that wide minima in the loss landscape generalize better than narrow minima. Moreover, many important parameters such as learning rate, batch size, etc. affects how the training approach traverses these landscapes. And examining loss landscapes using topological methods may provide us some insights to neural network training. And also existing studies have analyzed the loss landscape by reducing the dimension which may not capture full info about the loss landscape, so we will explore the landscape in high dimension itself.

So, the primary objectives of this project are:-

- To leverage topological methods in order to enhance our capacity to visualize and gain insights into the loss landscapes associated with training of deep neural networks.
- To explore the loss landscapes in high dimension so as to preserve more information.

So, our project aspires to shed light on the intricate topological structures that underlies the training dynamics of deep neural networks. By doing so, we aim to provide valuable insights that can inform the development of more robust and efficient training strategies, ultimately advancing the field of deep learning.

Chapter 2

Motivation

The paper [8] examines how the choice of initial learning rate affects the generalization performance of a neural network. It compares the outcomes when starting with two different initial learning rates: 0.1 and 0.01 using WideResnet-16. Interestingly, it finds that initially, the smaller learning rate of 0.01 yields better results, showing superior performance up to the 125th epoch as seen in 2.1 .

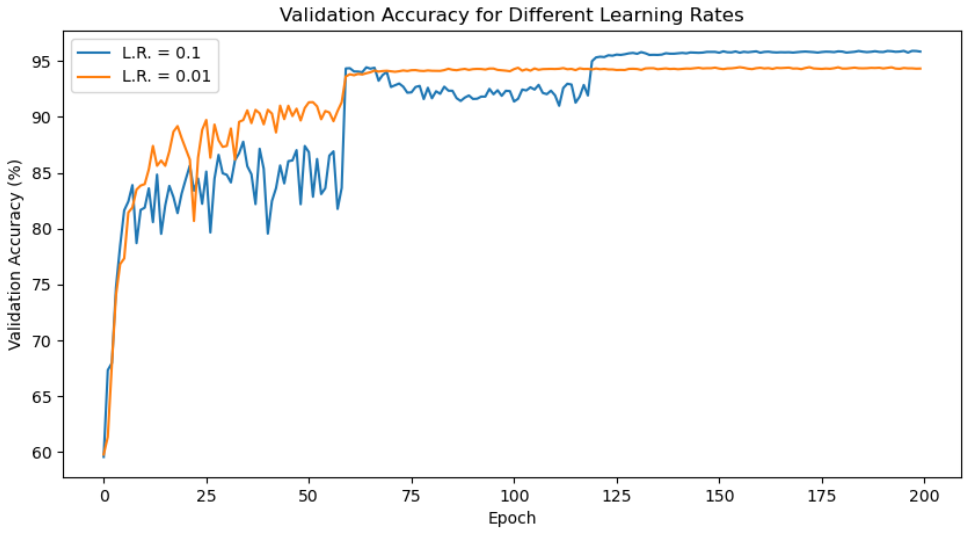

Figure 2.1: CIFAR-10 accuracy vs. epoch for WideResNet with weight decay, no data augmentation, and initial l.r. of 0.1 vs. 0.01 for validation set.

However, as training progresses, the larger initial learning rate of 0.1 eventually surpasses the smaller one. By the end of the 200th epoch, there is a noticeable difference in accuracy between the two cases. This shift in performance dynamics becomes more pronounced after

the learning rate is annealed three times, at the 60th, 125th, and 150th epochs. To understand why this discrepancy in generalization occurs, the study turns to topological analysis. By examining the topological features of the training process, such as changes in loss landscapes and the emergence of critical points, researchers aim to uncover the underlying reasons behind the observed differences in generalization performance.

Chapter 3

Theoretical Background

3.1 Neural Networks

Neural networks, inspired by the brain, use interconnected nodes to learn from data and make predictions. They're like puzzles, piecing together information to understand the whole picture. These networks are vital in many fields, from recognizing images and speech to driving innovation in complex problem-solving.

Hyperparameters shape a neural network's behavior and performance during training. Examples include the learning rate, which dictates optimization steps' size, and the network's depth and width. Tuning hyperparameters is crucial for optimal performance, akin to fine-tuning knobs on a machine to enhance accuracy and efficiency.

Optimizers are essential tools in training neural networks, refining parameters to minimize loss functions. Stochastic Gradient Descent (SGD) stands as a fundamental optimizer, updating parameters based on gradients calculated from a subset of training data. This iterative process guides the network toward optimal parameter values, enhancing learning and performance in the pursuit of improved accuracy and efficiency.

Loss functions are metrics used to evaluate how well a machine learning model performs on its training data. One common loss function is Cross Entropy, which measures the difference between predicted and actual values in classification tasks. Think of it like a scorecard: the lower the Cross Entropy, the better the model is at accurately predicting outcomes.

3.2 WideResnet

Wide Residual Networks (WRNs) are deep neural network architectures that aim to improve the performance of traditional Residual Networks (ResNets) by increasing model width. Introduced by Sergey Zagoruyko and Nikos Komodakis in 2016 [9], WRNs are characterized by their wider

convolutional layers compared to standard ResNets.

The key idea behind WRNs is to increase the number of channels in convolutional layers while keeping the network depth relatively shallow. This wider architecture helps to capture more diverse features and improve model expressiveness, leading to enhanced performance on various tasks such as image classification and object detection.

3.3 Contour Trees

A contour tree [5] is a topological data structure used in computational geometry for analyzing and representing the topology of scalar fields defined on geometric domains. Scalar fields associate a scalar value with each point in space, such as temperature on a surface or intensity in an image.

The contour tree captures the evolution of level sets in the scalar field as the scalar values change. A level set is a set of points in the domain where the scalar field has a constant value. The contour tree represents the relationships between these level sets as they merge and split.

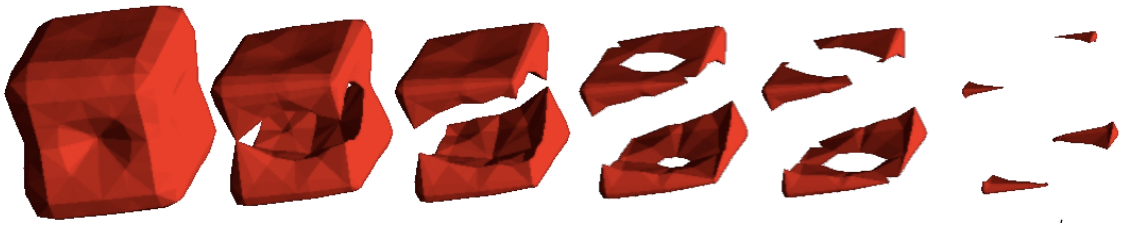

Figure 3.1: The number of contours of an isosurface changes as the isovalue is swept thought the function range

The contour tree is created by integrating both the **join tree** and **split tree** to provide a comprehensive understanding of the topological changes within the scalar field. The join tree captures the creation of connected components, and the split tree captures their dissolution. The contour tree, therefore, represents the birth and death of connected components as the scalar field parameter varies, offering insights into the complex structure of the data. It serves as a powerful tool for analyzing and visualizing topological features in various fields, including computational geometry and scientific visualization.

3.4 Persistence Diagram

The persistence diagram, introduced by Edelsbrunner, Letscher and Zomorodian [4], is a point set in the extended plane that encodes the difference in the homology of the sub-level sets of

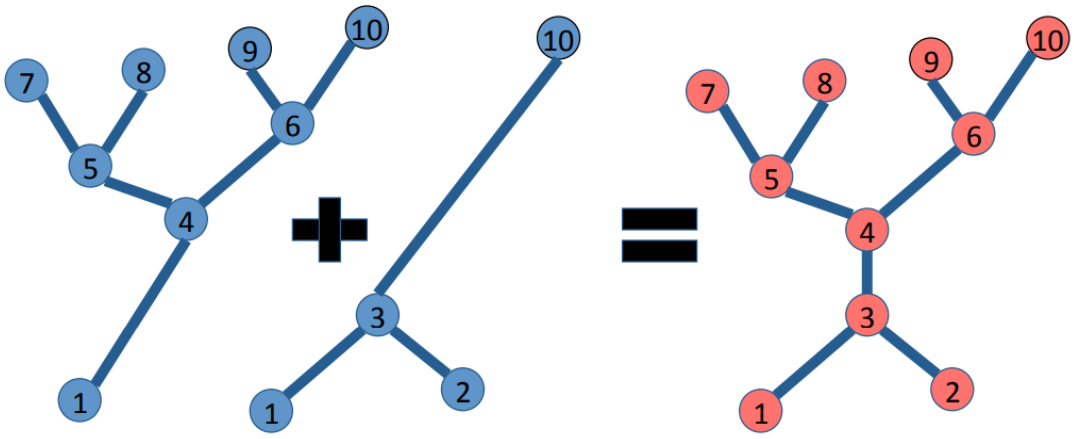

Figure 3.2: Corresponding join tree, split tree and contour tree

the function. Each point corresponds to a feature and quantifies its importance by the absolute difference between the point's two coordinates.

3.4.1 Total Persistence

Persistence [2], in the context of computational topology and persistent homology, refers to the characteristic of topological features in a dataset to persist or endure across different scales. It is a measure of how long these features, such as connected components, loops, or voids, exist during the process of filtration.

Here's a simple explanation:

Filtration: Imagine your dataset is like a three-dimensional landscape. Filtration is like looking at this landscape through a series of lenses, each revealing different details. As you switch between these lenses, the landscape appears differently, exposing various features.

Birth and Death: Features in the landscape, like hills or valleys, "come to life" (birth) and eventually "vanish" (death) as you move through the different lenses. Birth marks when a feature appears, and death is when it disappears.

Persistence Pair: Each feature has a pair of values associated with it – its birth and death. This pair tells you at which points in the filtration process the feature was born and when it disappeared.

Persistence Interval: The persistence interval for a feature is the time it exists, calculated as the difference between its death and birth values. It tells you how long the feature sticks around as you change your perspective.

Total Persistence: Total persistence is like adding up the "lifetimes" of all the interesting

things you discover in your dataset. It's a way to measure the overall activity or significance of various features in your data.

3.5 Wasserstein Distance

The Wasserstein distance [1] quantifies the dissimilarity between two probability distributions, a concept extended to persistence diagrams, aiding in the comparison of topological features across datasets.

The Wasserstein distance $W_p(f, g)$ between the persistence diagrams of functions f and g is defined as follows:

$$W_p(f,g) = \left(\sum_{\ell} \inf_{\gamma_\ell} \sum_{x \in \mathrm{Dgm}_\ell(f)} \|x - \gamma_\ell(x)\|_\infty^p \right)^{\frac{1}{p}},$$

where ℓ denotes each dimension, γ_ℓ represents all possible bijections between the persistence diagrams $\mathrm{Dgm}_\ell(f)$ and $\mathrm{Dgm}_\ell(g)$, and p is a parameter greater than the total persistence degree k .

3.6 Correlation with Neural Network Training:

Training Dynamics: The contour tree and total persistence can be computed at different stages of neural network training. This allows us to observe how the critical points and topological features evolve over time.

Correlation Analysis: By comparing the topological features with training performance metrics (e.g., accuracy, convergence speed), we can explore potential correlations. For instance, certain changes in the contour tree may align with critical points where the network experiences difficulties during training.

Chapter 4

Related Work

Many researchers have already looked into studying loss landscapes and connecting them with topological methods. They've explored this area, trying to understand how the shape of the loss landscape relates to the underlying structure revealed by topological analysis.

Paper [7] talks about how neural networks learn, specifically by examining the landscapes of their loss functions. Their focus is on finding better ways to navigate these complex landscapes during training, which is crucial for improving how neural networks perform tasks. Their study looks closely at how the structure of a network, the shape of the loss landscape, and certain training choices all work together. They aim to understand how these factors impact the learning process and the network's ability to generalize what it has learned.

This paper introduces a new method called "filter normalization" to help us see and understand the shape of the loss function. Imagine it as a tool that provides a clearer picture of the twists and turns in the landscape where the neural network is trying to find the best solutions. This fresh approach adds unique insights to the conversation about understanding these landscapes. The overall goal is to gather knowledge that can make training neural networks more effective and boost their overall performance.

- Paper [6] talks about connections between the generalization performance of neural networks and the geometry of their loss landscape near local minima. Visualizing this landscape is crucial, but existing methods are limited by linearity and only capture a few dimensions. Their study introduces a novel "jump and retrain" procedure, enhancing the sampling of the loss landscape. Non-linear dimensionality reduction through PHATE, combined with computational homology, allows meaningful visualization and quantification of differences between networks in terms of generalization.

In this paper, it's mentioned that neural network solutions, known as minima, which are flat in

the loss landscape tend to generalize better on diverse data. Additionally, the paper suggests that a good solution, or optimum, exhibits low topological changes in its surrounding landscape, indicating stability. This contrasts with less favorable solutions, where the landscape changes more dramatically. In simpler terms, a flatter and more stable terrain around the optimal solution contributes to better generalization in neural networks.

But the main drawback of [7] is that they are just generating 2 random direction and visualizing the loss landscape in those direction only. This could lead to loss of information and we could miss out on some critical information.

Chapter 5

Preliminary Work

In the initial phase of our project, we thoroughly examined various research papers and applied their methods to our investigations, aiming to uncover valuable insights into how neural networks learn. In our experimentation, we employed diverse model architectures, namely ResNet56, VGG9, and Densenet121, to conduct image classification on the CIFAR-10 dataset. This comprehensive approach allowed us to analyze and compare the behavior of different architectures throughout the training process. Utilizing techniques from various papers, we visualized the contours of loss landscapes at different training stages, offering a crucial visual representation for comprehending the evolution of neural networks.

An intriguing pattern in our observation surfaced as epochs increased, the optimal point exhibited low error, while the immediate neighborhood surrounding the optimal point experienced an increase in loss value as shown in figure 5.1,5.2,5.3 for VGG9 model. The similar behaviour is observed for Resnet56 and Densenet121 moels. This nuanced insight into the dynamics of neural networks during training adds depth to our understanding of their behavior and performance.

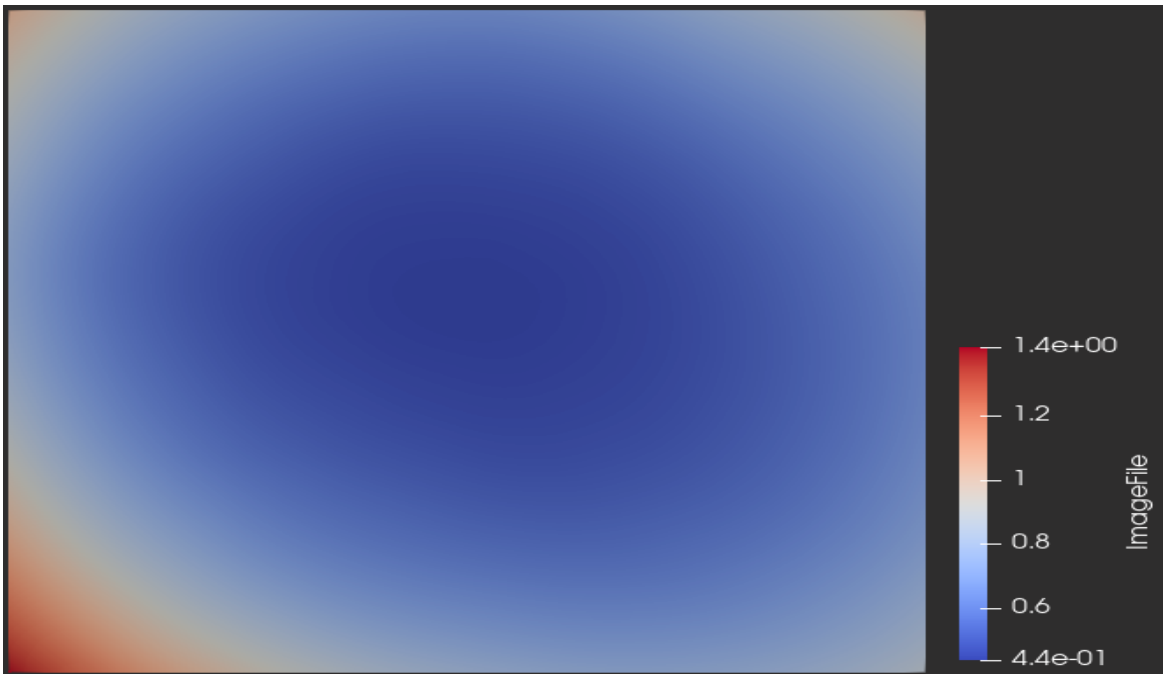

Figure 5.1: Neighbourhood Loss ranges at epoch 10(VGG-9).

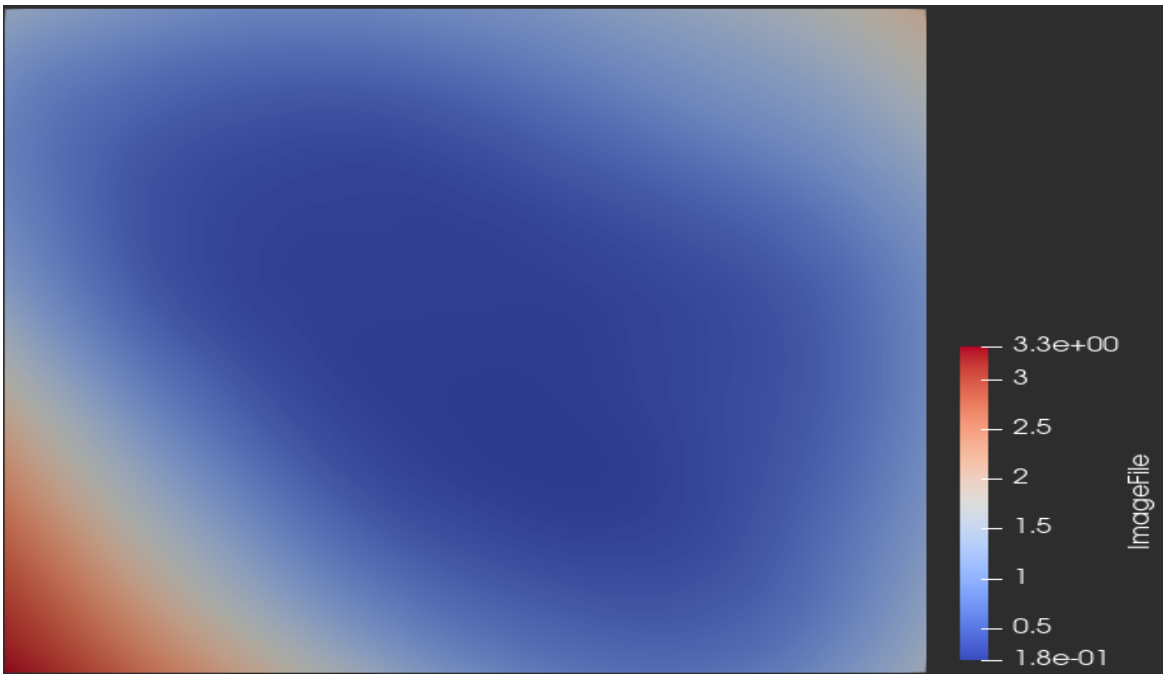

Figure 5.2: Neighbourhood Loss ranges at epoch 150(VGG-9).

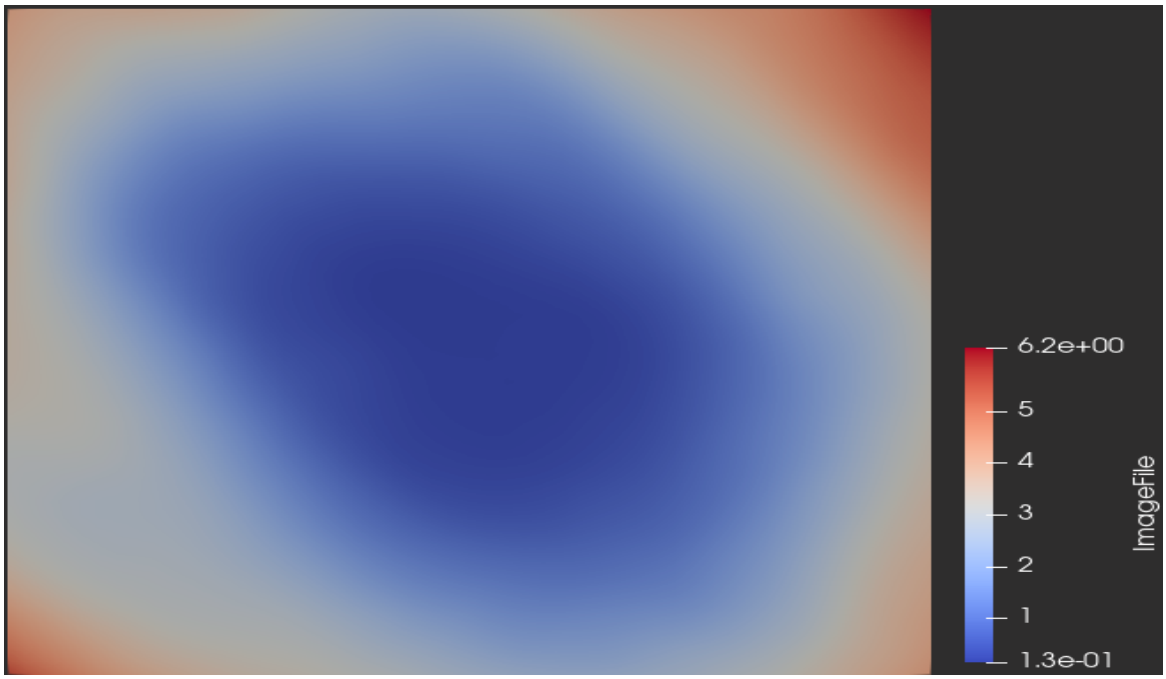

Figure 5.3: Neighbourhood Loss ranges at epoch 300(VGG-9).

Chapter 6

Methodology

Our method aims to address and overcome the limitations encountered by [7]. As in this approach, we are looking at loss landscape without reducing the dimension.

Figure 6.1 depicts our approach in multiple stages. These stages are defined in detailed:-

6.1 Computing Loss for each image

As our model learns over time, we notice that its performance, measured by loss values, fluctuates from one training epoch to the next. This average loss that is essentially an average of how well the model performs on each individual piece of data it's trained on.

For a single image x_i with corresponding label y_i, the loss $\ell(x_i, y_i)$ measures how much the model's prediction differs from the true label. In our case, we're using the CIFAR-10 dataset, which consists of 50,000 images for training and 10,000 for validation. So, to gauge how our model is doing, we calculate the training loss by averaging out the losses incurred on each of those 50,000 images:

$$\text{Train Loss} = \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \ell(x_i, y_i)$$

Similarly, we calculate the validation loss by averaging out the losses on the validation set, consisting of 10,000 images:

$$\text{Validation Loss} = \frac{1}{N_{\text{val}}} \sum_{i=1}^{N_{\text{val}}} \ell(x_i, y_i)$$

Here, N_{train} and N_{val} denote the number of images in the training and validation sets, respectively.

The loss function for each image, denoted as $\ell(x_i, y_i)$, depends on weight parameters corresponding to each neuron and features related to the image. Therefore, for each image, our loss function can be expressed as:

$$L : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}$$

where n is the number of neurons and d represents the parameters associated with the image.

Now at a particular epoch θ, the weights corresponding to neurons are constant. Thus, the loss function at θ can be represented as:

$$L_{\theta'} : \mathbb{R}^d \to \mathbb{R}$$

In this expression, θ' represents the parameters at epoch θ.

This comprehensive approach provides us with a deep insight into how our model is progressing, as we can analyze the performance of every single image across different training sessions.

6.2 Creating Nearest Neighbour Graph

On the other side we will be creating a nearest neighbor (NN) graph for both the training and validation sets.Since each image has a dimension of 28x28 pixels, resulting in 784 features, the Nearest Neighbor graph is constructed based on these features. So now we will create nearest neighbour for these images with k=20 using scikit-learn library. This graph serves as a visual representation of the relationships and similarities between the images, laying the groundwork for subsequent topological analyses.

It's a crucial step as it unveils the proximity of images to one another, forming the basis for constructing join and split trees that highlight topological features like peaks and valleys within the dataset. This NN graph remains constant throughout our analysis, providing a stable framework for evaluating loss values at each training epoch.

6.3 Constructing Merge Trees

Now, we're moving on to creating merge trees for each training epoch. These trees, called join tree and split tree, are derived from the nearest neighbor graph we built earlier. So we use

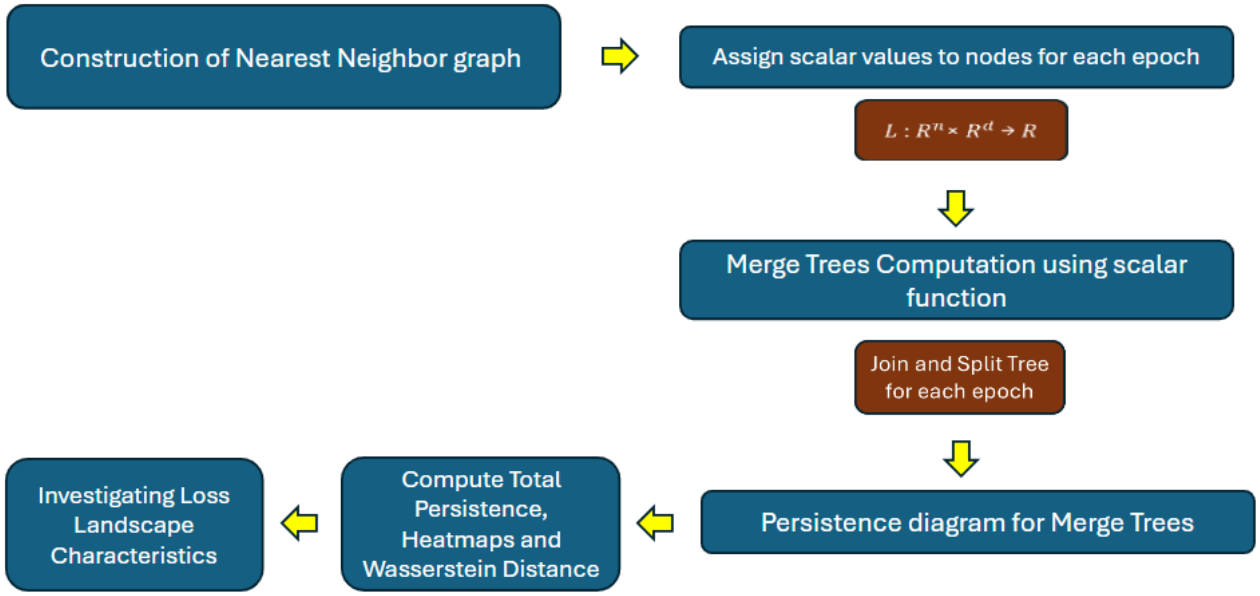

Figure 6.1: Methodology

the loss function $L_{\theta'}$ for images for constructing the merge trees (unique for each epoch). This means that as the neural network learns during training, it influences the structure of these trees.

We utilized an online code repository by Harish Doraiswamy for this purpose [3].

6.4 Constructing Persistence diagram

After constructing both the join and split trees, we computed persistence diagrams for each tree. These diagrams are crucial as they provide a visual representation of the lifespan of topological features present in the data. Imagine each feature, such as a connected component, as a living entity with a beginning (birth) and an end (death). The persistence diagram captures this lifecycle, plotting each feature as a point where its position on the diagram corresponds to its birth and death.

After computing the Persistence Diagram, we analysed them using total persistence, heatmaps and wasserstein distance.

6.4.1 Total Persistence

With the persistence diagram, we determine total persistence for each epoch. It's a measure that helps us understand the changes in the structure of our data.

6.4.2 Heatmaps for Persistence diagram

We have generated heatmaps for the persistence diagram to identify the predominant types of persistence, offering a richer insight compared to the persistence diagram alone. This visualization method allows for a more precise comparison of persistence diagrams, enabling us to discern subtle patterns and variations in topological features across datasets with greater clarity.

6.4.3 Wasserstein Distance

Subsequently, We employed the Wasserstein distance to quantify the differences between various persistence diagrams, aiming to compare them in a more robust manner. Leveraging the stability of the Wasserstein distance, this approach enables a more reliable comparison, facilitating a deeper understanding of the topological variations present in the datasets.

We followed a unique approach by comparing the persistence diagram of the 200th epoch with those of other epochs. This method provided valuable insights into how the topological features of the dataset evolved over the course of training, offering a nuanced understanding of the model's learning process.

Chapter 7

Results

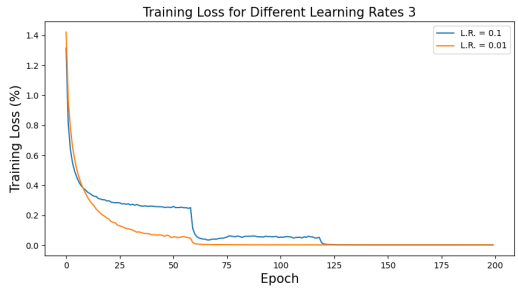

Figure 7.1: Training Loss for WideResNet-16 for 0.1 and 0.01 initial learning rates.

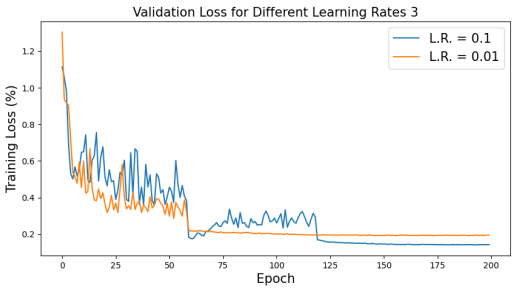

Figure 7.2: Validation Loss for WideResNet-16 for 0.1 and 0.01 initial learning rates.

In our study of CIFAR-10 training, we explore the loss landscape's topological features to understand optimization and generalization in deep neural networks. We plot average loss functions for training and validation sets, alongside employing novel topological analyses like total persistence, heatmap-based persistence diagrams, and Wasserstein distance. These methods unveil the landscape's evolution, critical point distribution, and structural changes, offering insights into optimization behavior, generalization, and model robustness.

7.1 Total Persistence

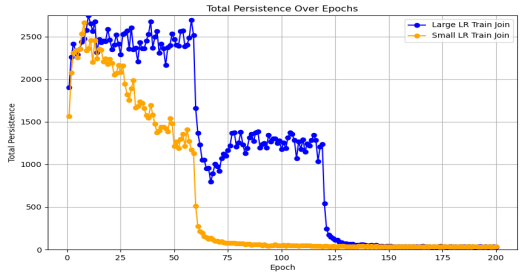

Figure 7.3: Total persistence comparison for Join Tree for training set.

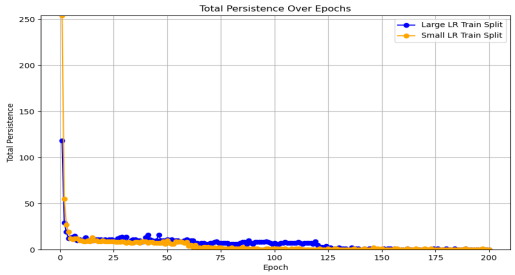

Figure 7.4: Total persistence comparison for Split Tree for training set.

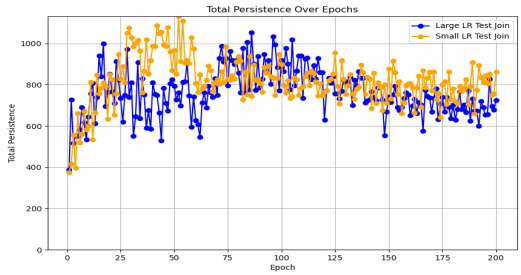

Figure 7.5: Total persistence comparison for Join Tree for validation set.

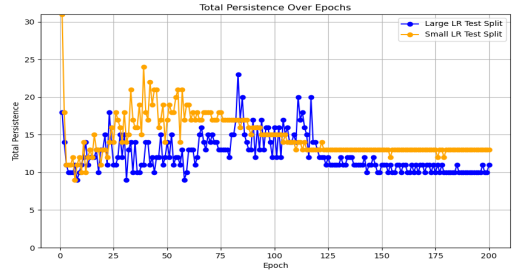

Figure 7.6: Total persistence comparison for Split Tree for Validation set.

When examining the total persistence diagrams for both training and validation sets, across both join and split trees, an intriguing pattern emerges. We observe a consistent decrease in total persistence across all four cases, mirroring the trend seen in average loss. This decline in total persistence signifies a notable transformation within the loss landscape over the course of training.

Essentially, as training progresses, we witness a reduction in the prominence of peaks and valleys within the landscape. This flattening of the landscape implies a smoothing-out effect, suggesting that the model is gradually converging towards a state characterized by fewer sharp fluctuations in loss values. In simpler terms, it indicates that the model is becoming more stable and less prone to large swings in performance as it learns from the data.

7.2 Heatmaps for Persistence Diagrams

Our analysis involves generating heatmaps for persistence diagrams at epochs 1, 59, 60, 124, 125, and 200, chosen for their alignment with pivotal shifts in learning rate at the 60th and 125th epochs. These epochs serve as critical junctures for observing how adjustments in learning rate shape the landscape's topological features. By focusing on these key transition points, our aim is to delve into the nuanced evolution of the landscape throughout the training process. This meticulous examination offers valuable insights into the dynamic interplay between learning rate variations and the intricate structure of the loss landscape in deep neural network training.

7.2.1 Training Set:-

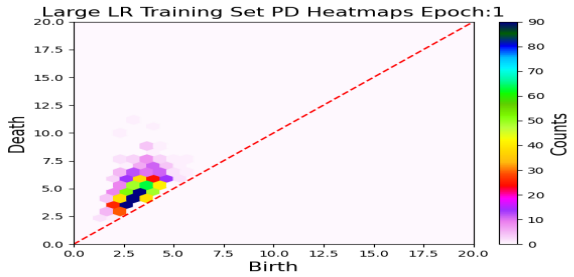

Figure 7.7: Heatmap for Large L.R.(epoch:1)

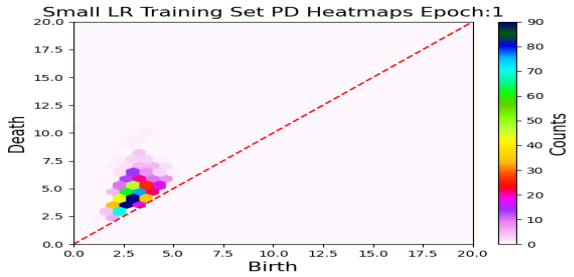

Figure 7.8: Heatmap for Small L.R.(epoch:1)

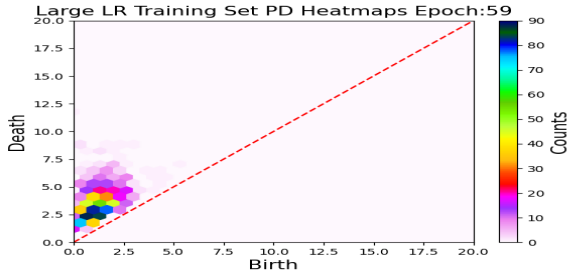

Figure 7.9: Heatmap for Large L.R.(epoch:59)

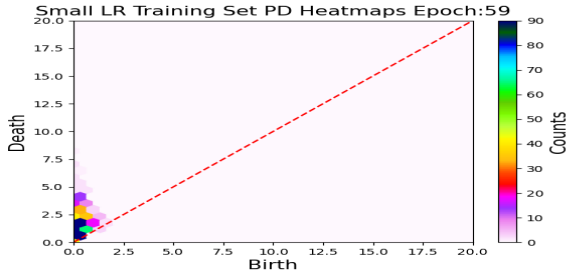

Figure 7.10: Heatmap for Small L.R.(epoch:59)

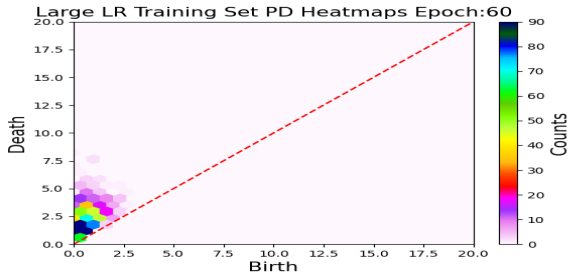

Figure 7.11: Heatmap for Large L.R.(epoch:60)

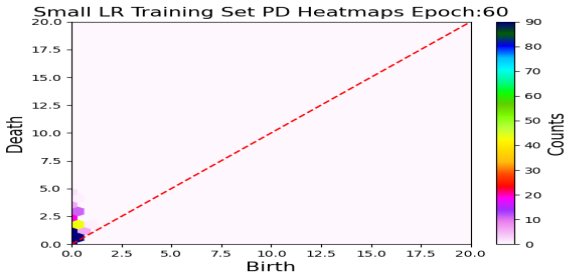

Figure 7.12: Heatmap for Small L.R.(epoch:60)

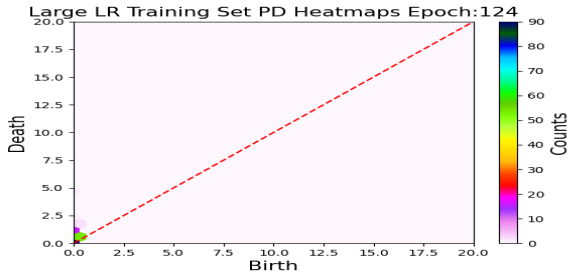

Figure 7.13: Heatmap for Large L.R.(epoch:124)

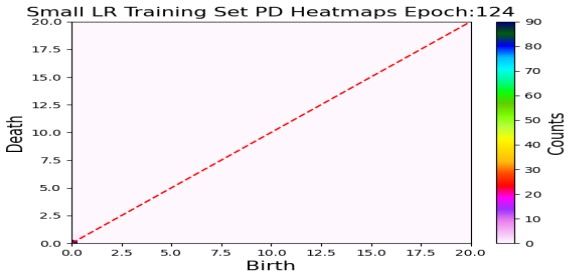

Figure 7.14: Heatmap for Small L.R.(epoch:124)

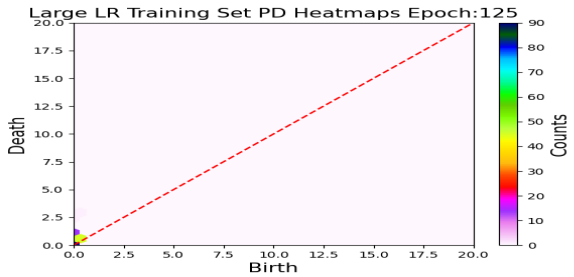

Figure 7.15: Heatmap for Large L.R.(epoch:125)

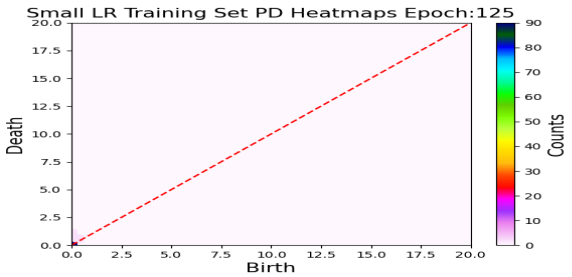

Figure 7.16: Heatmap for Small L.R.(epoch:125)

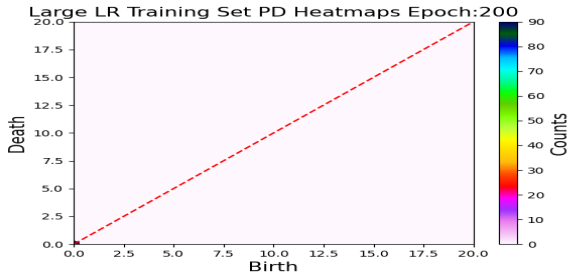

Figure 7.17: Heatmap for Large L.R.(epoch:200)

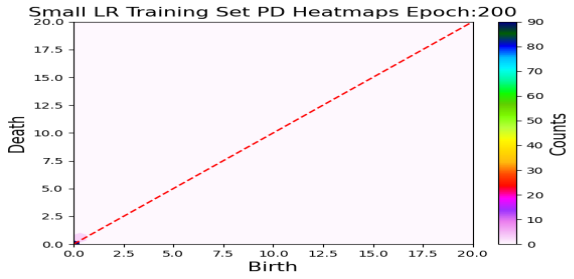

Figure 7.18: Heatmap for Small L.R.(epoch:200)

When examining the persistence diagram heatmaps for both large and small learning rates on the training set, distinct characteristics emerge. At the 200th epoch as shown in fig. 7.17 and fig. 7.18, both cases exhibit convergence towards the origin, suggesting a similar overall landscape at the end of training. However, a notable discrepancy arises at the 59th epoch(fig. 7.9 and fig. 7.10) and 60th epoch(fig. 7.11 and fig. 7.12): the small learning rate case demonstrates superior performance. This trend aligns with observations in the average loss plot, further corroborating the significance of learning rate choice.

From a structural perspective, these heatmaps reveal a gradual reduction in the steepness of peaks over the course of training. By the training's conclusion, the plots approach zero, indicating a flatter landscape. This transformation underscores the dynamic nature of the loss landscape and its sensitivity to changes in learning rate, ultimately shaping the model's performance and optimization trajectory.

7.2.2 Validation Set

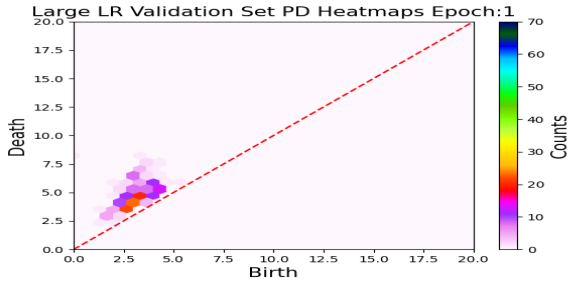

Figure 7.19: Heatmap for Large L.R.(epoch:1)

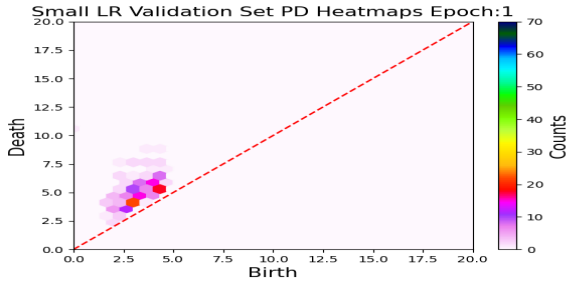

Figure 7.20: Heatmap for Small L.R.(epoch:1)

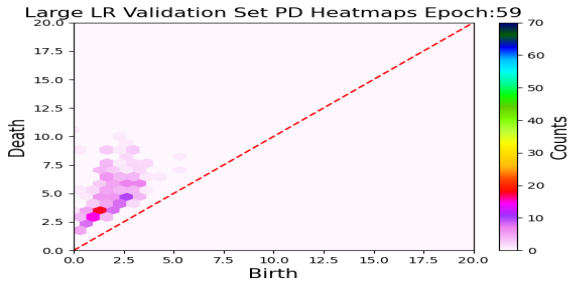

Figure 7.21: Heatmap for Large L.R.(epoch:59)

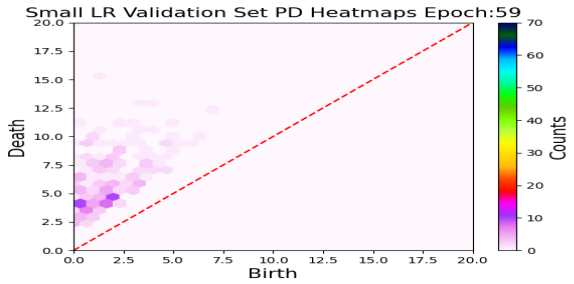

Figure 7.22: Heatmap for Small L.R.(epoch:59)

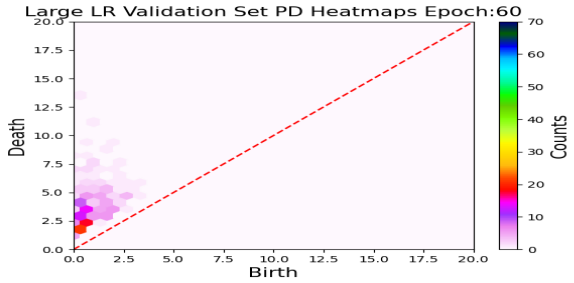

Figure 7.23: Heatmap for Large L.R.(epoch:60)

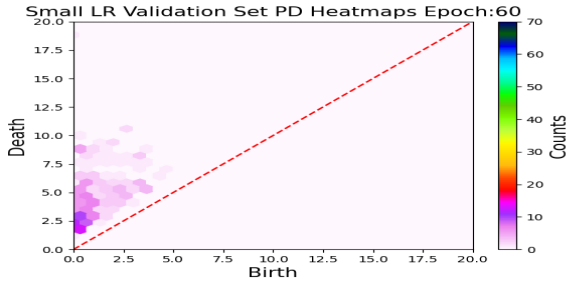

Figure 7.24: Heatmap for Small L.R.(epoch:60)

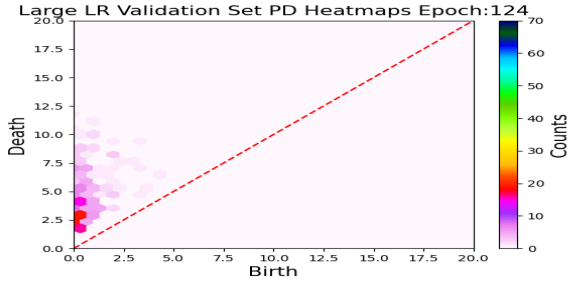

Figure 7.25: Heatmap for Large L.R.(epoch:124)

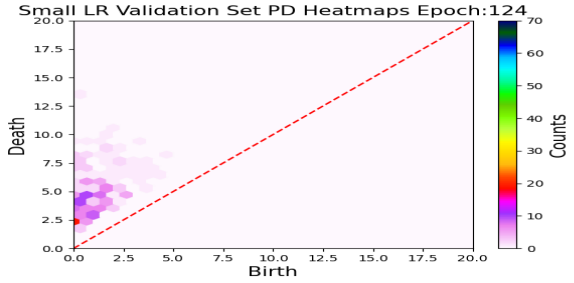

Figure 7.26: Heatmap for Small L.R.(epoch:124)

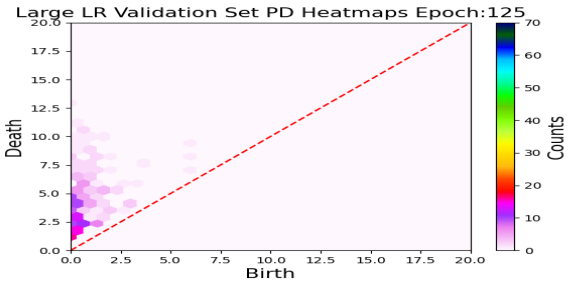

Figure 7.27: Heatmap for Large L.R.(epoch:125)

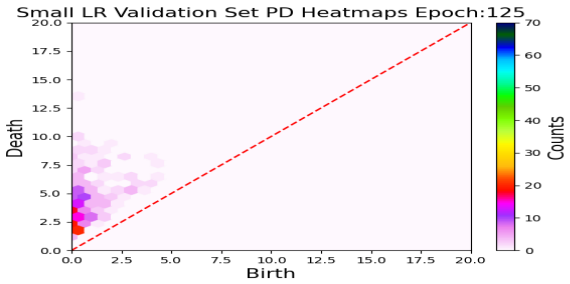

Figure 7.28: Heatmap for Small L.R.(epoch:125)

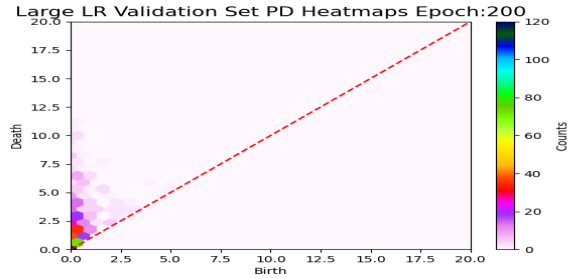

Figure 7.29: Heatmap for Large L.R.(epoch:200)

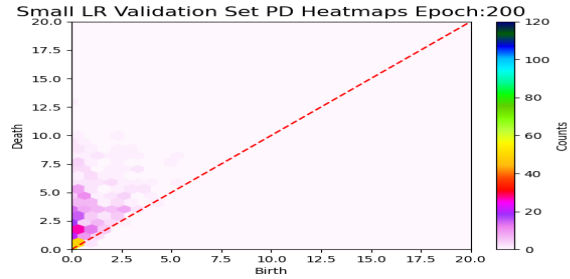

Figure 7.30: Heatmap for Small L.R.(epoch:200)

In assessing the validation set, it becomes apparent from the average loss plot that the large LR case outperforms the small LR case by the end of training. This trend is echoed in the total persistence analysis, affirming the superiority of the large LR approach. Furthermore, examination of the heatmap plot at the 200th epoch (fig. 7.29, fig. 7.30) reveals that the large LR case exhibits fewer high persistence points compared to the small LR case. This observation underscores the attainment of a flatter loss landscape by the large LR case, indicative of its enhanced optimization trajectory and superior generalization capabilities.

7.3 Wasserstein Distance

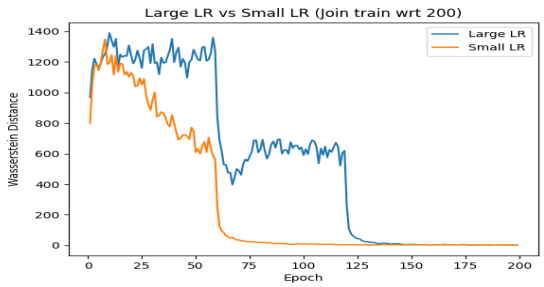

Figure 7.31: Wasserstein Distance for training set(Join Tree)

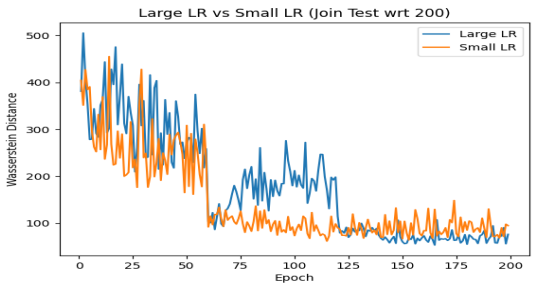

Figure 7.32: Wasserstein Distance for Validation set(Join Tree)

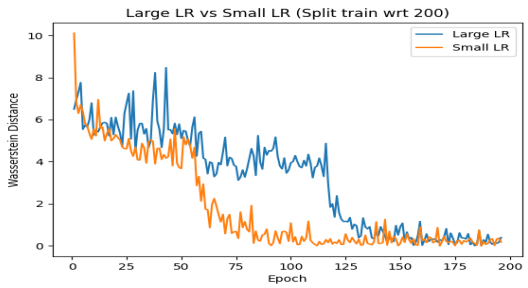

Figure 7.33: Wasserstein Distance for training set(Split Tree)

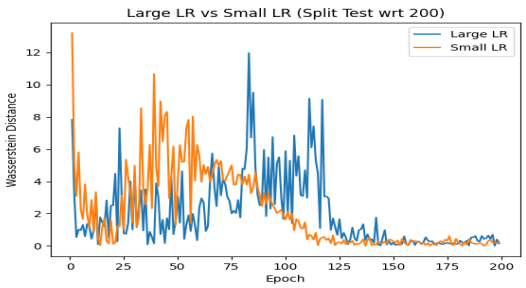

Figure 7.34: Wasserstein Distance for Validation set(Split Tree)

When we look at the Wasserstein distance (fig.7.31, fig. 7.32, fig. 7.33, fig. 7.34) we notice something interesting. It seems to move in sync with how well our model is doing in training and validation. Basically, when the Wasserstein distance changes, our model's accuracy and loss during training and validation also change. This suggests that there's a connection between how different our model's predictions are from the actual results and how well it's learning. So, keeping an eye on the Wasserstein distance can give us a good idea of how our model is performing overall.

Additionally, it's important to mention that by the 200th epoch, the loss landscape appears much smoother. This allows us to compare it to earlier epochs and see how noisy or smooth

the landscape was at different stages of training. This comparison helps us understand how the landscape changes over time and how that might affect how well the model performs.

7.4 Comparative Analysis with existing Approach:-

7.4.1 Heatmaps for ResNet-56 with and without skip connections:-

Our approach successfully distinguishes between loss landscapes for different learning rates. We further investigated whether it can also differentiate between the loss landscapes of ResNet-56 with and without skip connections.

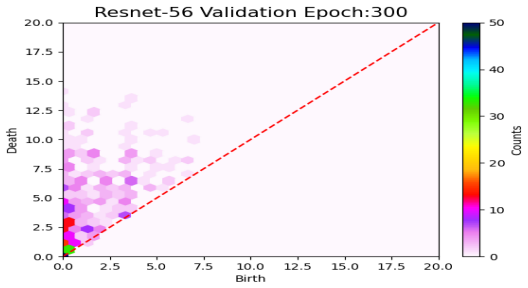

Figure 7.35: Heatmap for ResNet-56(Skip Connection) Epoch-300

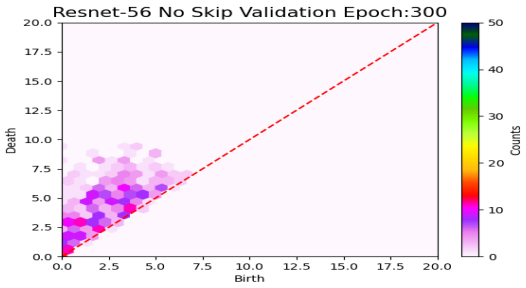

Figure 7.36: Heatmap for ResNet-56(No Skip Connection) Epoch-300

Upon examining these two persistence diagrams fig. 7.35 and fig. 7.36, it is evident that our method effectively captures the differences in the loss landscapes for these two scenarios. It is clearly evident that the skip connections case exhibits a greater number of low persistence maxima compared to the case without skip connections which have grater number of hih persistence maxima. This indicates that the loss landscape with skip connections is much smoother and flatter at the end of training compared to the loss landscape without skip connections.

7.4.2 Loss Landscape for different Learning rate with existing approach:-

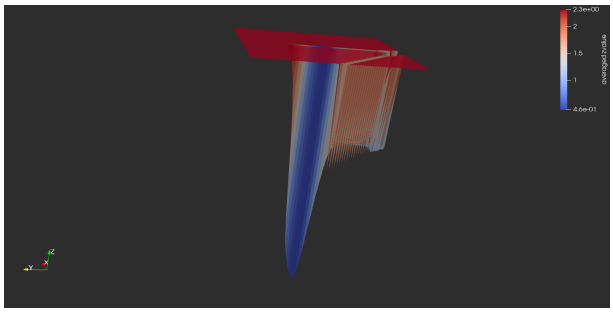

Figure 7.37: Loss landscape for Large LR(WideResNet epoch-200)

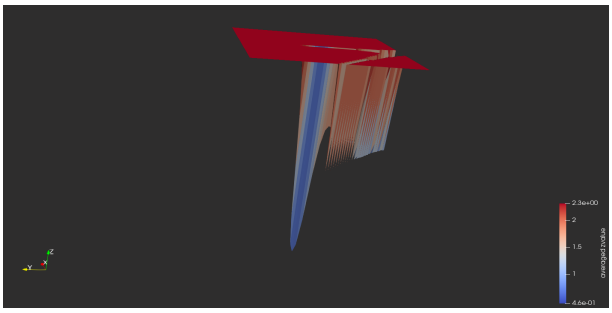

Figure 7.38: Loss landscape for Small LR(WideResNet epoch-200)

Upon examining these two landscapes fig. 7.37 and fig. 7.38, it is evident that there is no clear difference, indicating that the existing approach fails to capture the differences in the loss landscape for different initial learning rates. This shortcoming arises because the existing method examines only two random dimensions, leading to a loss of vital information.

In contrast, our approach successfully captures the differences in these loss landscapes, demonstrating its effectiveness in providing a more comprehensive analysis. Therefore, we can confidently say that our method is significantly better at differentiating loss landscapes.

7.5 Summary of Results:-

Analyzing total persistence, heatmaps for persistence diagrams, and Wasserstein distance provides valuable insights into our model's performance dynamics:

- Analyzing the loss landscape in high dimensions preserves more information compared to examining it in low dimensions.
- Topological properties correlate directly with model performance, aiding comprehensive understanding of its learning trajectory.
- They also serve as indicators of the smoothness or noise levels within the loss landscapes, offering crucial context regarding the optimization process's intricacies.

- The observed phenomenon aligns with established theories indicating that lower topological activities tend to enhance generalization, a notion further substantiated by comparing small and large LR cases.
- Notably, smoother and flatter loss landscapes emerge as key facilitators of improved generalization capabilities, underlining the significance of landscape structure in model performance optimization.

Chapter 8

Future Scope

In the next part of our project, we're going to dig deeper into the different things that happen when we train neural networks. Our main focus will be to carefully look into the important factors that play a role in training. Till now, we have only dealt with the learning rate case. This could be extend to others factors affecting the generalisation such as batch size, and dealing with issues like vanishing gradients, impact the training. We'll build on what we've already learned about Persistence Diagrams, and we're also going to explore other interesting concepts related to shapes and structures in the data. This phase is all about getting a better understanding of the complex processes that govern how neural networks learn.

The final goal should be to discover novel phenomenon related to topological analysis of neural network loss landscapes.

Chapter 9

Conclusion

Exploring loss landscapes, which depict how neural networks learn, can lead to exciting new findings. By applying topological analysis to these landscapes, we get a unique perspective on how neural networks undergo training. Our experiments, combined with insights from other researchers, reveal a clear trend: as training continues, the level of topological changes decreases.

Comparing large and small learning rates using total persistence, heatmaps, and Wasserstein distance reveals distinct loss landscape shapes. **Large LR yields smoother landscapes**, aiding better generalization. **Small LR fosters rugged terrains with pronounced peaks**, potentially hindering adaptability. Understanding these nuances is crucial for optimizing machine learning models.

The experiments corroborate the phenomenon wherein **reduced topological activity correlates with improved generalization**. This observation aligns with the broader understanding that a flatter, less complex loss landscape tends to facilitate better generalization in machine learning models.

Analyzing the **loss landscape in high dimensions preserves more information** compared to examining it in low dimensions. Since our approach operates in high dimensions, it effectively captures the differences in loss landscapes for various cases, outperforming existing methods.

Bibliography

[1] Wasserstein distance. https://en.wikipedia.org/wiki/Wasserstein_metric. 7

[2] Mathieu Carriere, Frédéric Chazal, Marc Glisse, Yuichi Ike, Hariprasad Kannan, and Yuhei Umeda. Optimizing persistent homology based functions. pages 1294–1303, 2021. 6

[3] Harish D. Contour tree github repository. https://github.com/harishd10/contour-tree. 15

[4] Edelsbrunner, Letscher, and Zomorodian. Topological persistence and simplification. *Discrete & computational geometry*, 28:511–533, 2002. 5

[5] Han4Wei'Shen. Contour tree. https://web.cse.ohio-state.edu/~shen.94/788/Site/Slides_files/contourTree.pdf. 5

[6] S. Horoi, J. Huang, B. Rieck, G. Lajoie, G. Wolf, and S. Krishnaswamy. Exploring the geometry and topology of neural network loss landscapes. *Neural Computation*, 25:67–89, 2022. 8

[7] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018. 8, 9, 13

[8] Yuanzhi Li, Colin Wei, and Tengyu Ma. Towards explaining the regularization effect of initial large learning rate in training neural networks. *Advances in neural information processing systems*, 32, 2019. 2

[9] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 4