COL780 A2:- PatchCamelyon Image Classification

Yash Bansal (2022CS51133)

Contents

1 Introduction

The PatchCamelyon (PCam) dataset comprises approximately 327,680 color images of histopathology scans from lymph node sections. Each image has a resolution of 96 $\times$ 96 pixels. The task is a binary classification problem, where the goal is to determine whether a given tissue sample contains metastatic tissue.

The dataset is well-balanced, with approximately equal numbers of positive and negative examples in the training, validation, and test sets.

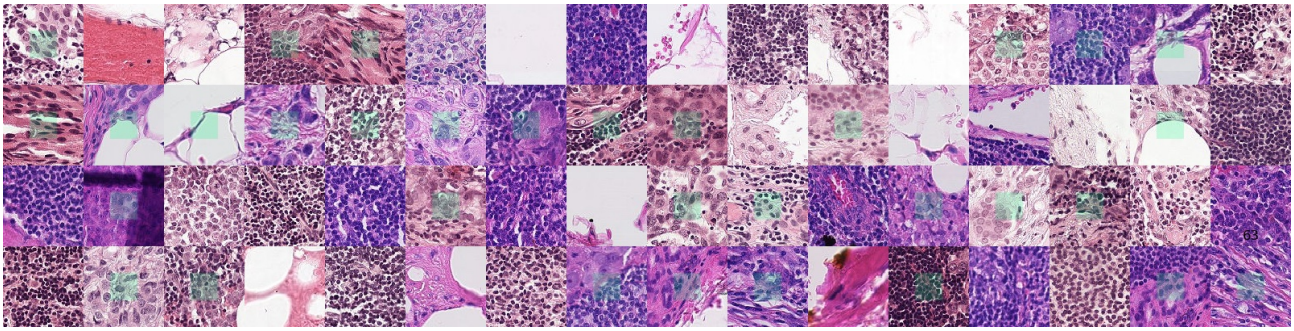


Figure 1: Sample dataset images

For our study, we conduct ablation experiments using ResNet, VGG, and a custom architecture incorporating both residual and inception blocks. The dataset is split into training, validation, and test sets in an 80-10-10 ratio. We analyze the impact of various hyperparameters and architectural choices by training models for 25 epochs and plotting loss and accuracy curves for both training and validation sets. Additionally, we report test accuracy at the 10th, 20th, and 25th epochs, along with the best achieved test accuracy.

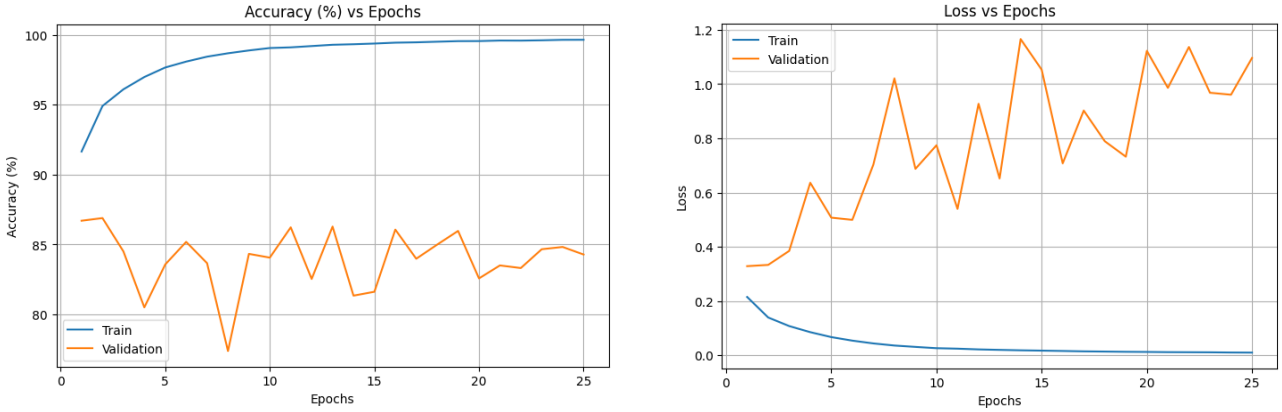
All models are implemented in Python using PyTorch, with pretrained ResNet and VGG models loaded from the `torchvision` library.

The trained models are saved in this link: **SharePoint Link**

2 Ablation studies on Resnet

2.1 Base Resnet18 Model

- Learning rate:- 0.001
- Optimizer:- Adams

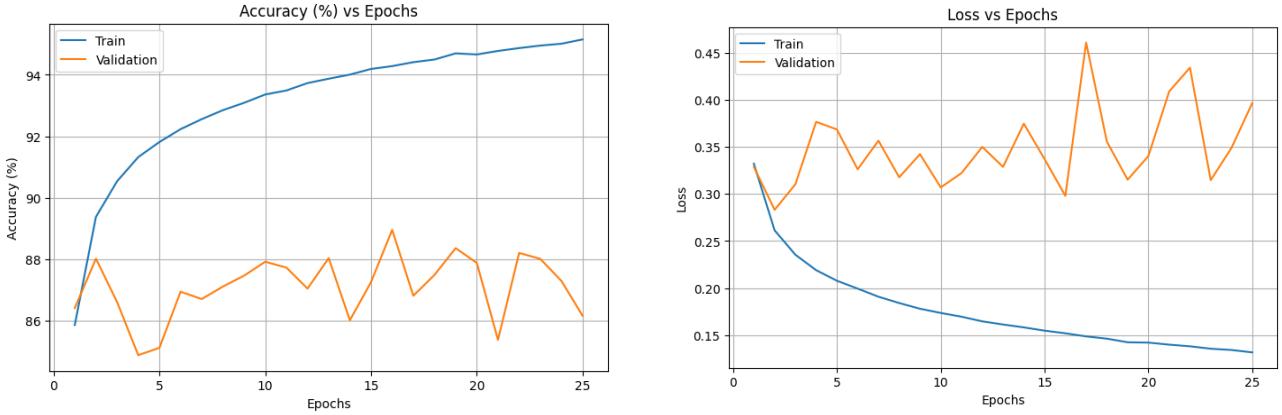


(a) Accuracies (b) Losses

Figure 2: Resnet18 Model without Data Augmentation

In the basic ResNet-18 model without any data augmentation, we observe significant overfitting. The training loss approaches zero, and accuracy reaches nearly 100%, while the validation loss increases with epochs. To mitigate this overfitting, we apply data augmentation techniques such as random horizontal and vertical flips, color jittering, random cropping, and random rotations of up to 15 degrees. These augmentations enhance the model's generalization ability, reducing overfitting and improving its performance on unseen data.

2.2 Augmentation Techniques



(a) Accuracies (b) Losses

Figure 3: Resnet18 Model with Data Augmentation Techniques

Set	Metric	No Data Augmentation			Data Augmentation		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	99.06	99.55	99.65	93.36	94.66	95.15
	Precision	0.99	1.00	1.00	0.94	0.95	0.96
	Recall	0.99	1.00	1.00	0.92	0.94	0.94
	F1 Score	0.99	1.00	1.00	0.93	0.95	0.95
Validation	Accuracy (%)	84.06	82.57	84.28	87.92	87.88	86.16
	Precision	0.95	0.96	0.95	0.90	0.96	0.97
	Recall	0.72	0.68	0.73	0.85	0.79	0.75
	F1 Score	0.82	0.80	0.82	0.88	0.87	0.84
Test	Accuracy (%)	81.525	80.054	81.137	86.667	84.757	83.011
	Precision	0.955	0.964	0.953	0.915	0.960	0.972
	Recall	0.662	0.624	0.655	0.808	0.725	0.680
	F1 Score	0.782	0.758	0.776	0.858	0.826	0.800
Max Accuracy on Test Set (%)		**85.202**			**87.1094**		

Table 1: Comparison between no augmentation and augmentation for Resnet18

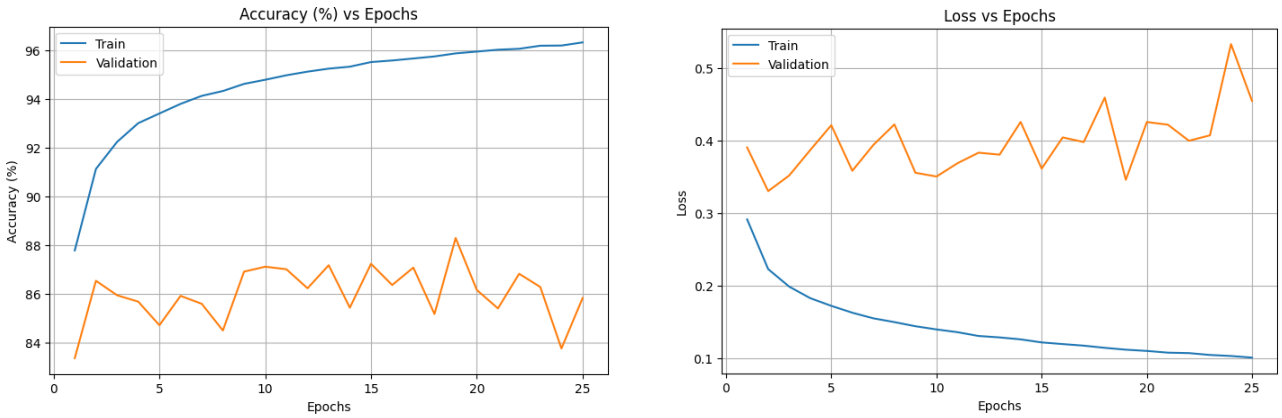
As observed in the loss and accuracy graphs, the validation loss increases significantly over epochs when no data augmentation is applied. This occurs because the model memorizes the training data instead of learning generalizable features, leading to poor performance on unseen data.

However, after applying augmentation, the validation loss stabilizes over epochs. This is because data augmentation introduces variations in the training data, preventing the model from overfitting to specific patterns. Additionally, the training loss no longer approaches zero, further indicating reduced overfitting, as the model is learning more robust and generalizable representations rather than simply memorizing the training samples.

Moreover, the best final testing accuracy improves by approximately 2% with data augmentation. This improvement is due to the model being exposed to diverse transformations of the data during training, making it more resilient to variations in real-world test samples.

2.3 Optimizers

2.3.1 SGD Optimizer

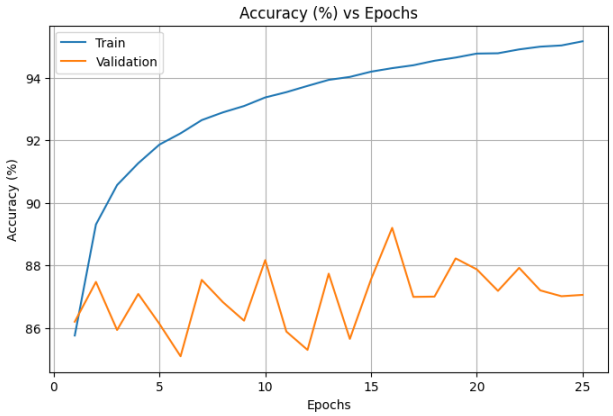


(a) Accuracies

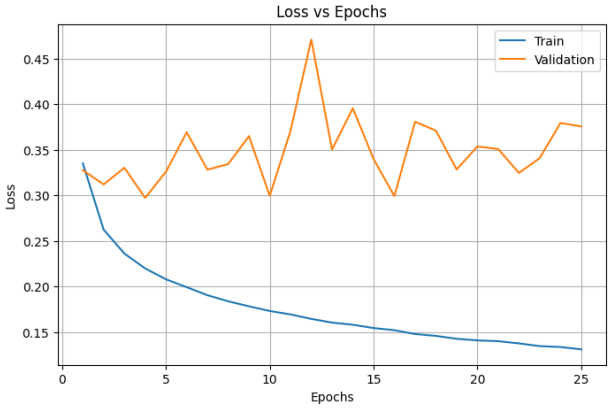
(b) Losses

Figure 4: Resnet18 with Stochastic Gradient Descent Optimizer

2.3.2 AdamW Optimizer



(a) Accuracies



(b) Losses

Figure 5: Resnet18 with AdamW optimizer

Set	Metric	SGD			AdamW		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	94.80	95.96	96.34	93.37	94.77	95.16
	Precision	0.95	0.97	0.97	0.94	0.96	0.96
	Recall	0.94	0.95	0.96	0.92	0.94	0.94
	F1 Score	0.95	0.96	0.96	0.93	0.95	0.95
Validation	Accuracy (%)	87.13	86.17	85.84	88.17	87.88	87.06
	Precision	0.94	0.94	0.96	0.90	0.96	0.96
	Recall	0.79	0.77	0.75	0.86	0.79	0.78
	F1 Score	0.86	0.85	0.84	0.88	0.87	0.86
Test	Accuracy (%)	83.823	83.881	81.787	87.122	84.680	84.497
	Precision	0.950	0.952	0.963	0.918	0.964	0.961
	Recall	0.714	0.713	0.661	0.815	0.720	0.719
	F1 Score	0.815	0.816	0.784	0.864	0.825	0.823
Max Accuracy on Test Set (%)		**86.0321**			**87.1216**		

Table 2: Comparison between different optimizers for Resnet18

During training with the Adam optimizer, we observed significant fluctuations in the loss curves, and the model continued to struggle with overfitting. To address this, we experimented with the SGD and AdamW optimizers.

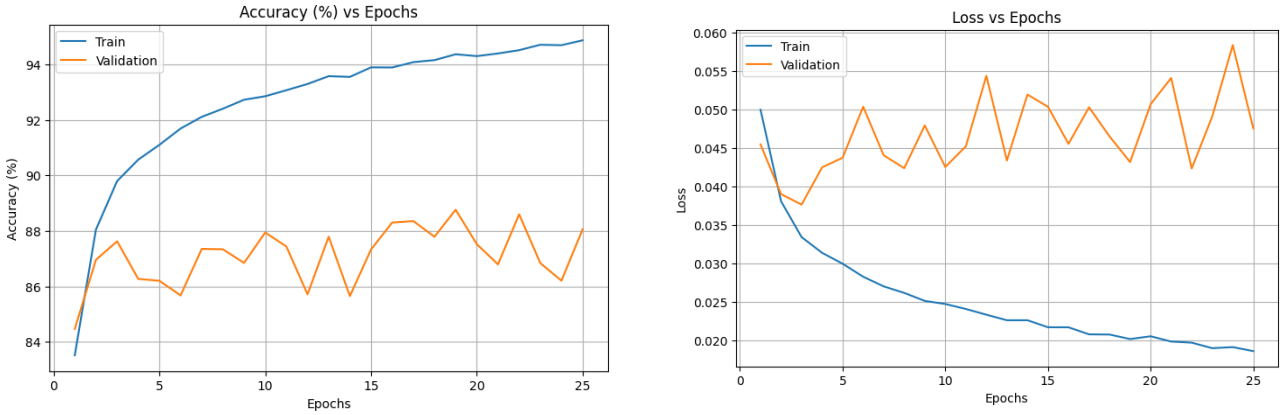
Applying the SGD optimizer reduced the fluctuations in training but came at the cost of approximately 1% lower accuracy. This is because SGD updates the model weights more conservatively, leading to a smoother optimization trajectory but potentially converging to a slightly less optimal solution.

On the other hand, the AdamW optimizer effectively reduced fluctuations without compromising accuracy. This is because AdamW modifies the weight decay mechanism, preventing the model from overfitting while maintaining stable and efficient updates.

Based on these observations, we choose AdamW as the optimizer for further experiments, as it balances stability and generalization without sacrificing accuracy.

2.4 Loss Functions

2.4.1 Focal Loss



(a) Accuracies

(b) Losses

Figure 6: Resnet18 with Focal Loss

Set	Metric	Cross-Entropy Loss 10	Cross-Entropy Loss 20	Cross-Entropy Loss 25	Focal loss 10	Focal loss 20	Focal loss 25
Train	Accuracy (%)	93.36	94.66	95.15	92.85	94.30	94.87
	Precision	0.94	0.95	0.96	0.94	0.95	0.96
	Recall	0.92	0.94	0.94	0.92	0.93	0.94
	F1 Score	0.93	0.95	0.95	0.93	0.94	0.95
Validation	Accuracy (%)	87.92	87.88	86.16	87.94	87.52	88.06
	Precision	0.90	0.96	0.97	0.93	0.96	0.96
	Recall	0.85	0.79	0.75	0.83	0.79	0.79
	F1 Score	0.88	0.87	0.84	0.87	0.86	0.87
Test	Accuracy (%)	86.667	84.757	83.011	85.956	84.924	85.327
	Precision	0.915	0.960	0.972	0.940	0.964	0.966
	Recall	0.808	0.725	0.680	0.768	0.726	0.732
	F1 Score	0.858	0.826	0.800	0.845	0.828	0.833
Max Accuracy on Test Set (%)		**87.1094**			**87.8082**		

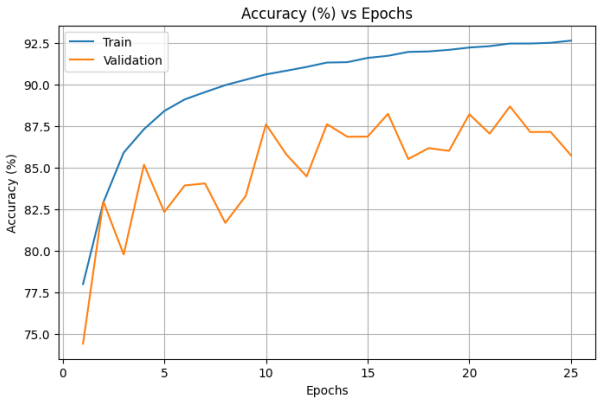
Table 3: Comparison between Cross-Entropy Loss and Focal Loss for Resnet18

Using focal loss does not significantly impact the model's training process or final results. This is because the dataset is well-balanced between positive and negative examples, meaning there is no severe class imbalance for focal loss to address.

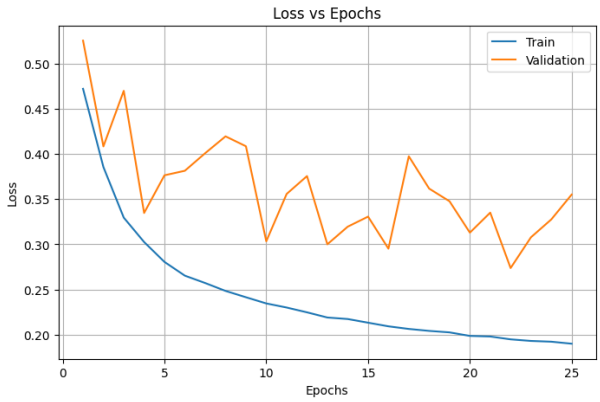
Focal loss is primarily designed to down-weight easy examples and focus more on hard-to-classify samples, which is beneficial in highly imbalanced datasets where the model tends to be biased toward the majority class. However, in a balanced dataset, focal loss behaves similarly to standard cross-entropy loss, as the modulating factor has little effect when class distributions are nearly equal.

Thus, in this scenario, focal loss does not provide any noticeable advantage over cross-entropy loss.

2.5 Learning Rate

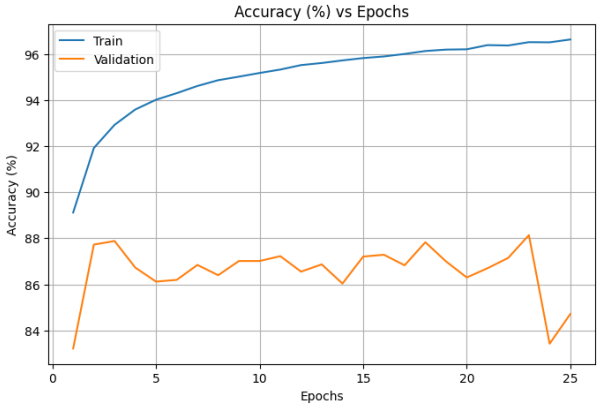


(a) Accuracies

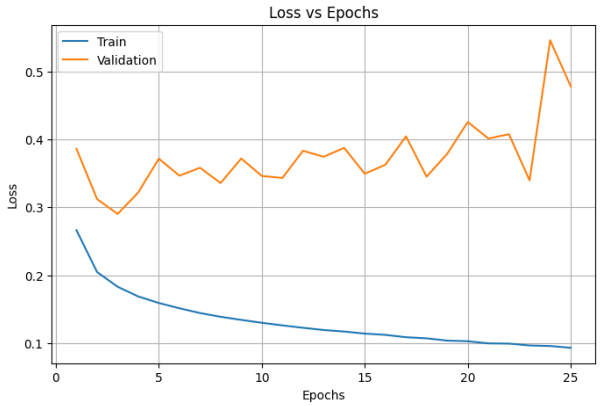


(b) Losses

Figure 7: Learning rate $1e-2$

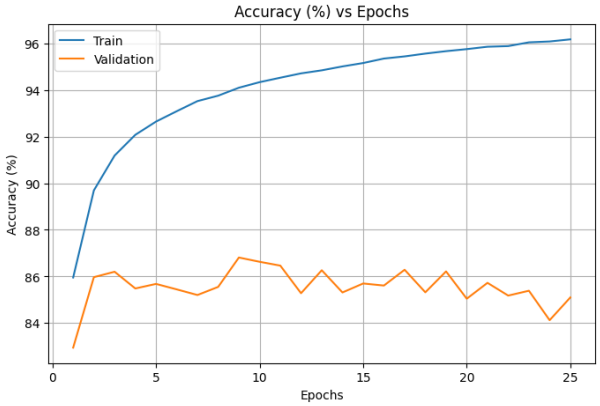


(a) Accuracies

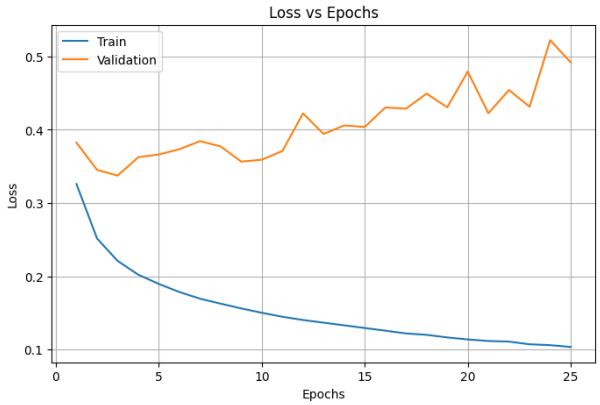


(b) Losses

Figure 8: Learning rate $1e-4$



(a) Accuracies



(b) Losses

Figure 9: Learning rate $1e-5$

Set	Metric	Learning Rate $1e-2$			Learning Rate $1e-3$		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	90.60	92.21	92.63	93.36	94.66	95.15
	Precision	0.91	0.93	0.93	0.94	0.95	0.96
	Recall	0.90	0.91	0.92	0.92	0.94	0.94
	F1 Score	0.91	0.92	0.93	0.93	0.95	0.95
Validation	Accuracy (%)	87.59	88.20	85.73	87.92	87.88	86.16
	Precision	0.89	0.94	0.94	0.90	0.96	0.97
	Recall	0.85	0.82	0.76	0.85	0.79	0.75
	F1 Score	0.87	0.87	0.84	0.88	0.87	0.84
Test	Accuracy (%)	85.858	85.034	83.676	86.667	84.757	83.011
	Precision	0.906	0.943	0.951	0.915	0.960	0.972
	Recall	0.800	0.746	0.710	0.808	0.725	0.680
	F1 Score	0.850	0.833	0.813	0.858	0.826	0.800
Max Accuracy on Test Set (%)		**85.8856**			**87.1094**		

Table 4: Comparison Between Learning Rates for Resnet18

Set	Metric	Learning Rate $1e-4$			Learning Rate $1e-5$		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	95.17	96.20	96.62	94.34	95.75	96.17
	Precision	0.96	0.97	0.97	0.95	0.96	0.97
	Recall	0.94	0.96	0.96	0.94	0.95	0.96
	F1 Score	0.95	0.96	0.97	0.94	0.96	0.96
Validation	Accuracy (%)	87.01	86.30	84.71	86.63	85.05	85.10
	Precision	0.94	0.95	0.97	0.94	0.94	0.95
	Recall	0.79	0.76	0.72	0.79	0.75	0.74
	F1 Score	0.86	0.85	0.82	0.85	0.83	0.83
Test	Accuracy (%)	84.052	83.966	80.808	83.023	82.803	82.153
	Precision	0.948	0.960	0.973	0.941	0.948	0.956
	Recall	0.721	0.709	0.634	0.705	0.694	0.674
	F1 Score	0.819	0.816	0.767	0.806	0.801	0.791
Max Accuracy on Test Set (%)		**85.9558**			**84.7595**		

Table 5: Comparison Between Learning Rates for Resnet18

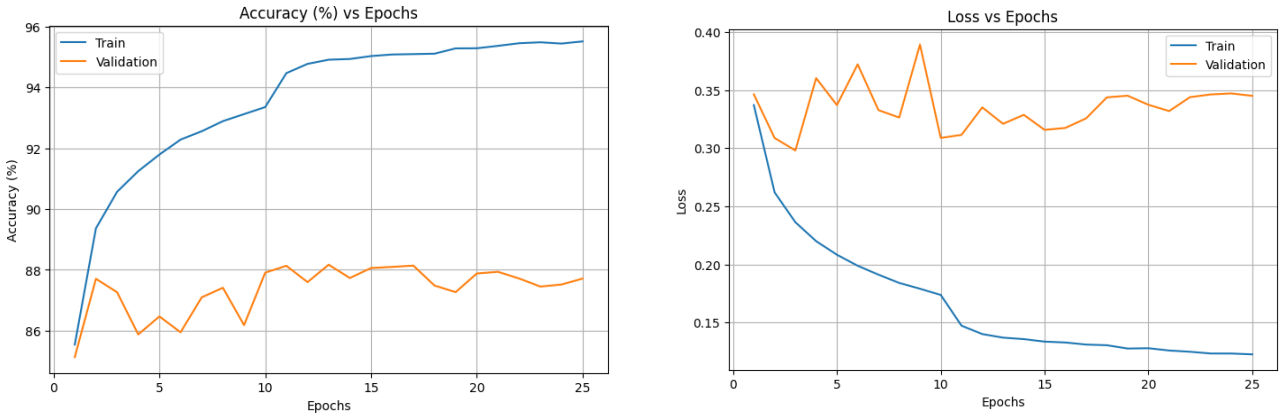
A learning rate of $1e-3$ proved to be the most effective for training ResNet-18. Other learning rates negatively impacted the training process, either by causing excessive fluctuations or by failing to learn meaningful features within the given number of epochs.

A higher learning rate, such as $1e-2$, led to unstable training dynamics, preventing the model from converging properly and resulting in a final accuracy drop of approximately 1.5%. Conversely, using a lower learning rate slowed down convergence, limiting the model's ability to learn effective feature representations within the given epochs, leading to an accuracy decrease of around 2%.

Thus, $1e-3$ strikes the best balance between stability and convergence speed, making it the optimal choice for further experiments.

2.6 Learning Rate Scheduling

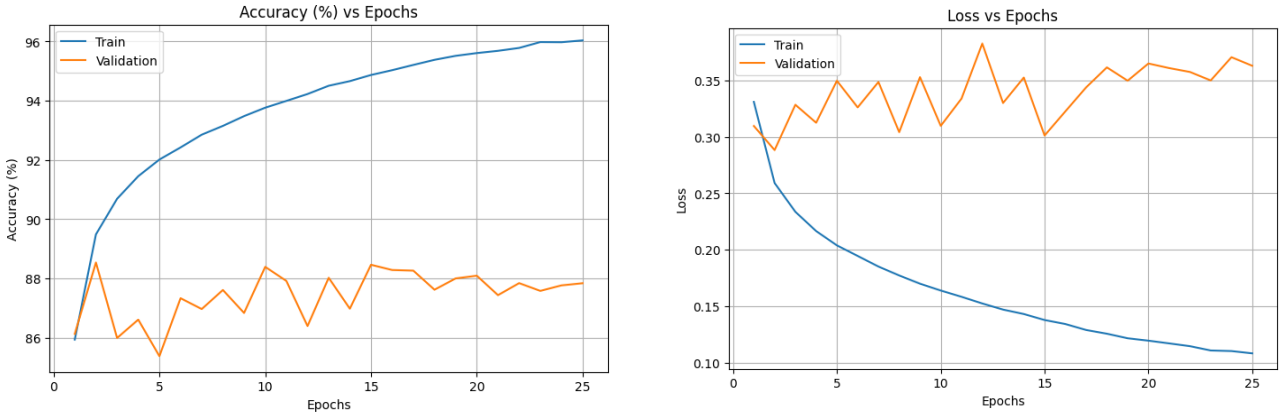
2.6.1 StepLR Scheduling



(a) Accuracies (b) Losses

Figure 10: StepLR Scheduling for Resnet18

2.6.2 Cosine Annealing LR scheduling



(a) Accuracies (b) Losses

Figure 11: Cosine Annealing LR Scheduling for Resnet18

Both StepLR and Cosine Annealing learning rate schedules effectively stabilize the model's training. In the later epochs, the reduced learning rate ensures smoother optimization, preventing large fluctuations in validation loss.

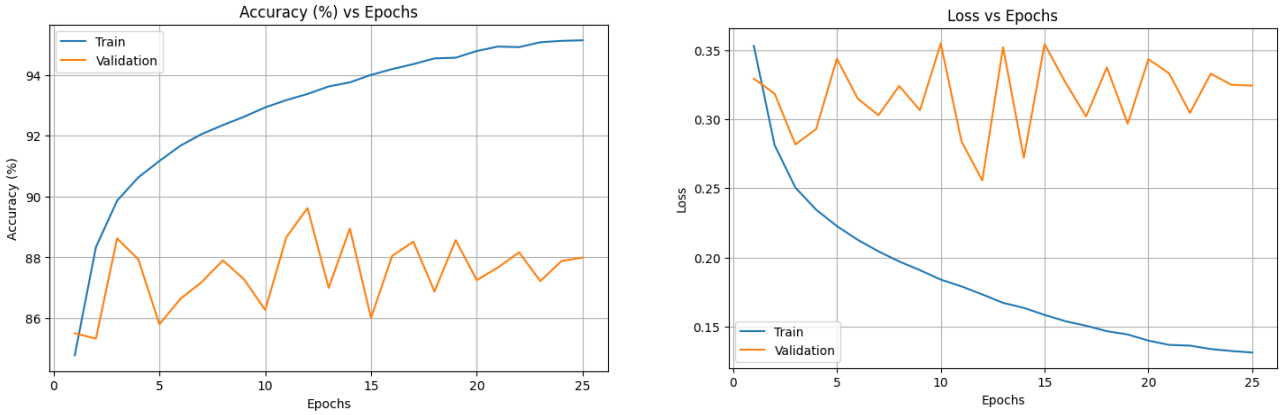
While the final accuracy remains nearly the same for both schedulers, the learning curve with Cosine Annealing is more stable throughout training. This smooth transition in learning rates helps the model converge more steadily without abrupt changes.

Based on these observations, we choose Cosine Annealing for further experiments, as it provides a more stable and gradual learning process.

Set	Metric	StepLR			Cosine LR		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	93.35	95.29	95.52	93.77	95.61	96.04
	Precision	0.94	0.96	0.96	0.95	0.96	0.97
	Recall	0.92	0.94	0.95	0.93	0.95	0.95
	F1 Score	0.93	0.95	0.95	0.94	0.96	0.96
Validation	Accuracy (%)	87.91	87.88	87.71	88.39	88.09	87.84
	Precision	0.93	0.95	0.95	0.92	0.96	0.96
	Recall	0.82	0.80	0.80	0.84	0.80	0.79
	F1 Score	0.87	0.87	0.87	0.88	0.87	0.87
Test	Accuracy (%)	87.174	85.599	85.693	86.343	85.904	84.894
	Precision	0.945	0.954	0.957	0.934	0.964	0.968
	Recall	0.789	0.748	0.747	0.782	0.746	0.722
	F1 Score	0.860	0.839	0.839	0.851	0.841	0.827
Max Accuracy on Test Set (%)		**87.3566**			**86.9354**		

Table 6: Comparison Between different Scheduling Techniques for Resnet18

2.7 Removing skip connections



(a) Accuracies

(b) Losses

Figure 12: Removing Skip Connections from Resnet18

Removing the skip connections from ResNet-18 results in a 1.5% increase in the final best testing accuracy. However, it significantly disrupts the learning process, leading to high fluctuations in validation loss and accuracy.

Skip connections play a crucial role in stabilizing deep network training by allowing gradients to flow more effectively through the layers, mitigating the vanishing gradient problem. Without them, the network struggles to learn smooth and consistent representations, causing unstable training dynamics. Although the final accuracy improves slightly in this case, the lack of stability makes the training process less reliable and harder to tune.

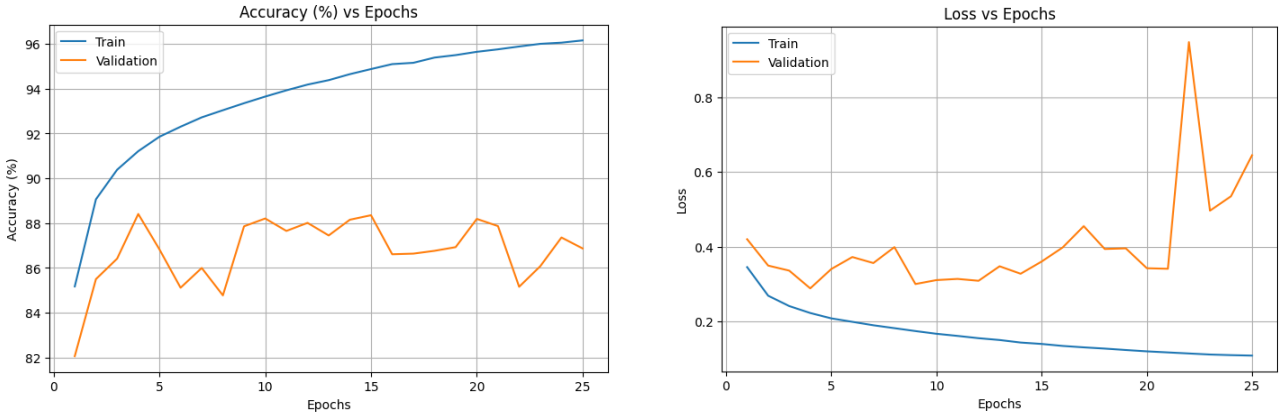
Therefore, we choose to retain skip connections in ResNet-18 to ensure a more stable and well-behaved training process.

Set	Metric	With Skip Conn. 10	With Skip Conn. 20	With Skip Conn. 25	Without Skip Conn. 10	Without Skip Conn. 20	Without Skip Conn. 25
Train	Accuracy (%)	93.36	94.66	95.15	92.94	94.79	95.14
	Precision	0.94	0.95	0.96	0.94	0.96	0.96
	Recall	0.92	0.94	0.94	0.92	0.94	0.94
	F1 Score	0.93	0.95	0.95	0.93	0.95	0.95
Validation	Accuracy (%)	87.92	87.88	86.16	86.27	87.25	87.99
	Precision	0.90	0.96	0.97	0.96	0.96	0.95
	Recall	0.85	0.79	0.75	0.76	0.78	0.80
	F1 Score	0.88	0.87	0.84	0.85	0.86	0.87
Test	Accuracy (%)	86.667	84.757	83.011	84.430	85.425	86.481
	Precision	0.915	0.960	0.972	0.960	0.965	0.956
	Recall	0.808	0.725	0.680	0.718	0.735	0.765
	F1 Score	0.858	0.826	0.800	0.822	0.835	0.850
Max Accuracy on Test Set (%)		**87.1094**			**88.4918**		

Table 7: Comparison between with and without skip connections in Resnet18

2.8 Number of Layers

2.8.1 Resnet34

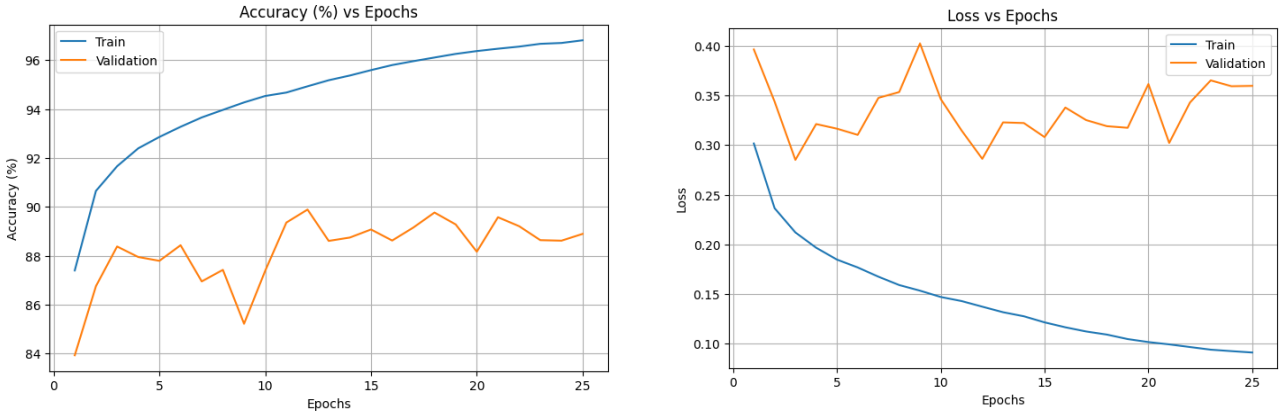


(a) Accuracies

(b) Losses

Figure 13: Resnet34

2.8.2 Resnet50



(a) Accuracies

(b) Losses

Figure 14: Resnet50

Set	Metric	Resnet34 10	Resnet34 20	Resnet34 25	Resnet50 10	Resnet50 20	Resnet50 25
Train	Accuracy (%)	93.65	95.65	96.16	94.53	96.37	96.81
	Precision	0.95	0.96	0.97	0.95	0.97	0.97
	Recall	0.93	0.95	0.95	0.94	0.96	0.96
	F1 Score	0.94	0.96	0.96	0.94	0.96	0.97
Validation	Accuracy (%)	88.21	88.19	86.87	87.39	88.17	88.89
	Precision	0.95	0.96	0.96	0.96	0.96	0.96
	Recall	0.81	0.80	0.77	0.78	0.79	0.81
	F1 Score	0.87	0.87	0.85	0.86	0.87	0.88
Test	Accuracy (%)	85.913	86.234	84.650	84.955	85.025	86.291
	Precision	0.956	0.962	0.963	0.962	0.969	0.969
	Recall	0.753	0.754	0.720	0.728	0.724	0.750
	F1 Score	0.842	0.846	0.824	0.829	0.829	0.845
Max Accuracy on Test Set (%)		**87.6404**			**87.8113**		

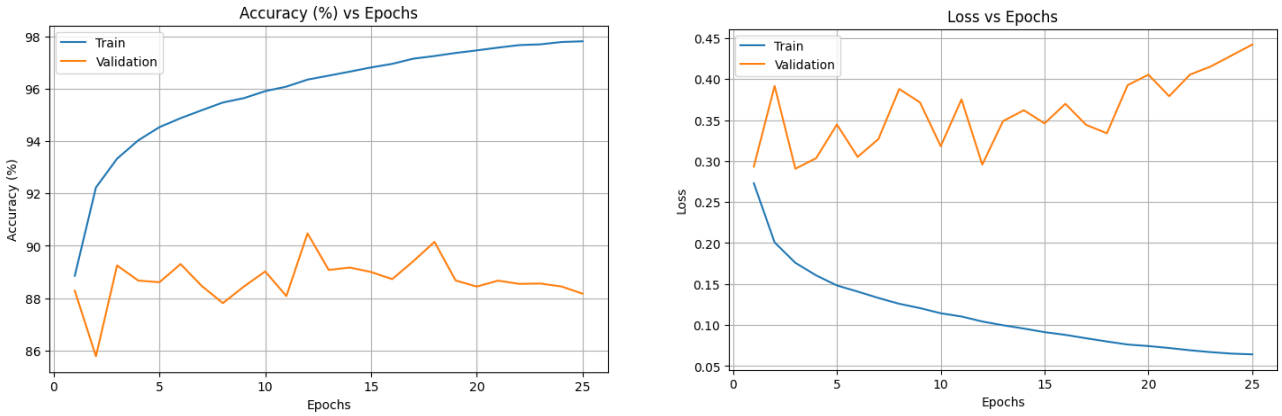
Table 8: Comparison Between Number of Layers in Resnet

Both ResNet-34 and ResNet-50 perform well, exhibiting significantly fewer fluctuations in training compared to ResNet-18. This improved stability is due to their deeper architectures, which allow for better feature extraction and more effective gradient propagation.

Among them, ResNet-50 provides the best overall performance, likely due to its increased depth and use of bottleneck layers, which enhance representation learning without excessively increasing computational cost.

Thus, we select ResNet-50 as the best ResNet model for our experiments.

2.9 Input Image Size



(a) Accuracies

(b) Losses

Figure 15: Upscaling Input Images to 224 × 224

Set	Metric	96 × 96			224 × 224		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	94.53	96.37	96.81	95.90	97.46	97.81
	Precision	0.95	0.97	0.97	0.97	0.98	0.98
	Recall	0.94	0.96	0.96	0.95	0.97	0.97
	F1 Score	0.94	0.96	0.97	0.96	0.97	0.98
Validation	Accuracy (%)	87.39	88.17	88.89	89.02	88.45	88.17
	Precision	0.96	0.96	0.96	0.97	0.98	0.98
	Recall	0.78	0.79	0.81	0.81	0.79	0.78
	F1 Score	0.86	0.87	0.88	0.88	0.87	0.87
Test	Accuracy (%)	84.955	85.025	86.291	86.920	86.099	86.395
	Precision	0.962	0.969	0.969	0.974	0.981	0.984
	Recall	0.728	0.724	0.750	0.759	0.736	0.740
	F1 Score	0.829	0.829	0.845	0.853	0.841	0.845
Max Accuracy on Test Set (%)		**87.8113**			**89.4562**		

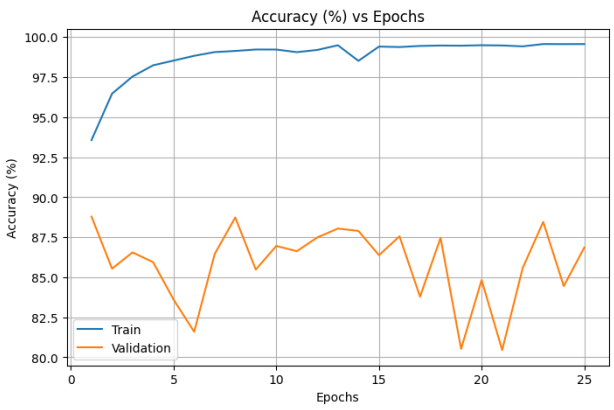
Table 9: Comparison Betweeen Input Images Sizes

Increasing the image size improves the accuracy of ResNet-50 by 2%. This is because we are using a pretrained model, originally trained on 224 × 224 images. As a result, the model has already learned optimal feature representations for this resolution, leading to better generalization and improved performance.

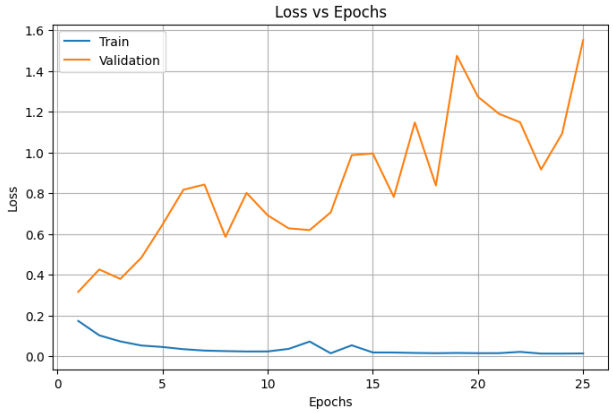
However, a larger image size significantly increases training time due to the higher computational cost associated with processing more pixels per image. Despite the accuracy improvement, this trade-off between accuracy and training efficiency must be considered when selecting the optimal image size.

3 Ablation studies on VGG

3.1 Base VGG16 Model



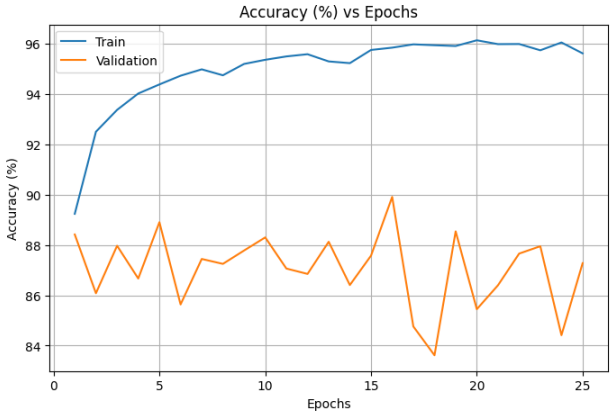
(a) Accuracies



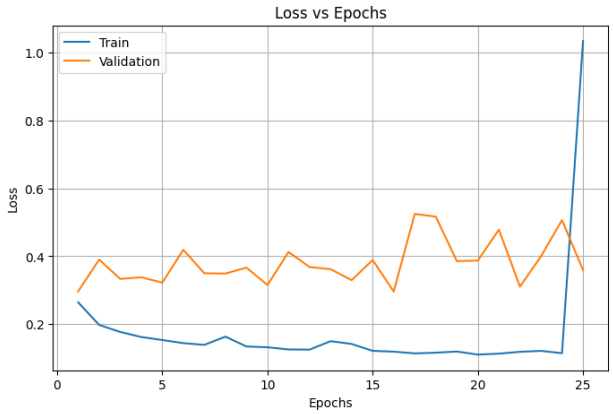
(b) Losses

Figure 16: VGG16 Model without Data Augmentation

3.2 Augmentation Techniques



(a) Accuracies



(b) Losses

Figure 17: VGG16 Model with Data Augmentation Techniques

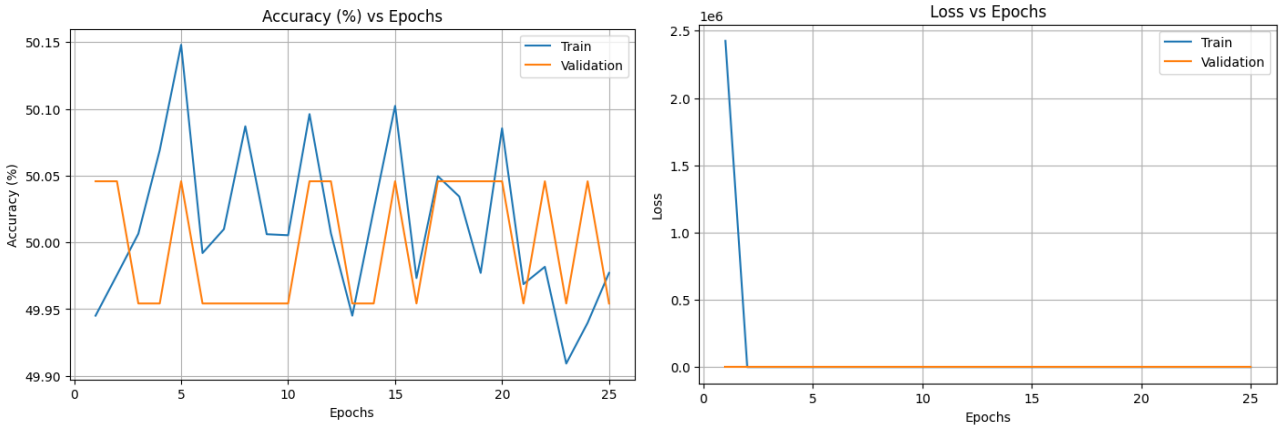
Using the same logic as in ResNet-18, data augmentation techniques improve the accuracy of VGG-16 by approximately 3% while reducing overfitting to the training examples.

Data augmentation introduces variations in the training data, preventing the model from memorizing specific patterns and encouraging it to learn more generalizable features. This helps improve performance on unseen data while making the model more robust. Additionally, augmentation reduces the gap between training and validation performance, mitigating overfitting and leading to a more stable learning process.

Set	Metric	No Data Augmentation			Data Augmentation		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	99.23	99.49	99.57	95.37	96.15	95.63
	Precision	0.99	0.99	1.00	0.96	0.97	0.96
	Recall	0.99	1.00	1.00	0.95	0.95	0.95
	F1 Score	0.99	0.99	1.00	0.95	0.96	0.96
Validation	Accuracy (%)	86.96	84.83	86.86	88.31	85.45	87.28
	Precision	0.96	0.97	0.98	0.96	0.94	0.96
	Recall	0.77	0.72	0.76	0.80	0.76	0.78
	F1 Score	0.85	0.82	0.85	0.87	0.84	0.86
Test	Accuracy (%)	83.588	81.979	83.261	86.655	84.430	86.365
	Precision	0.965	0.972	0.973	0.962	0.954	0.967
	Recall	0.697	0.658	0.684	0.763	0.723	0.753
	F1 Score	0.809	0.785	0.803	0.851	0.823	0.847
Max Accuracy on Test Set (%)		**87.1765**			**89.8712**		

Table 10: Comparison between no augmentation and augmentation for VGG16

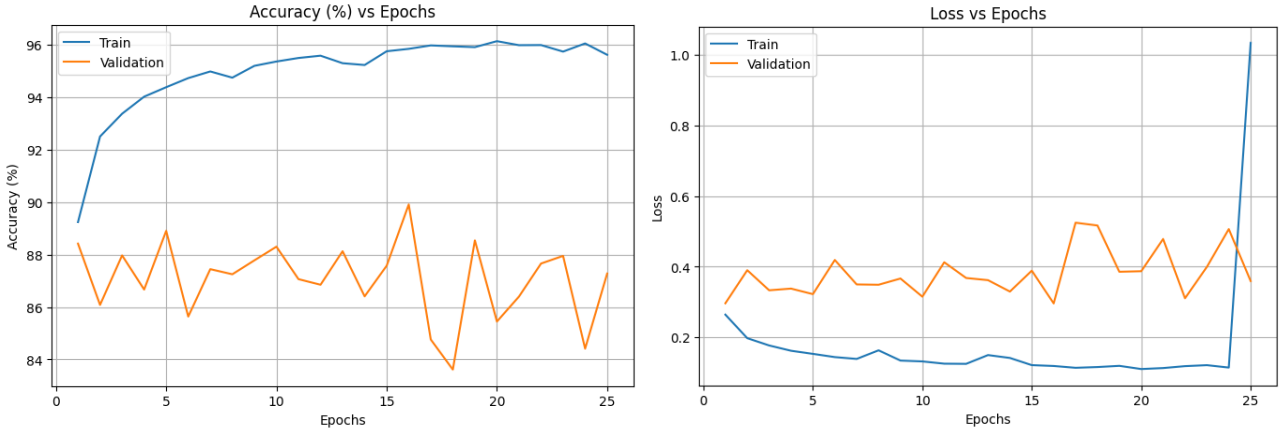
3.3 Learning Rate



(a) Accuracies

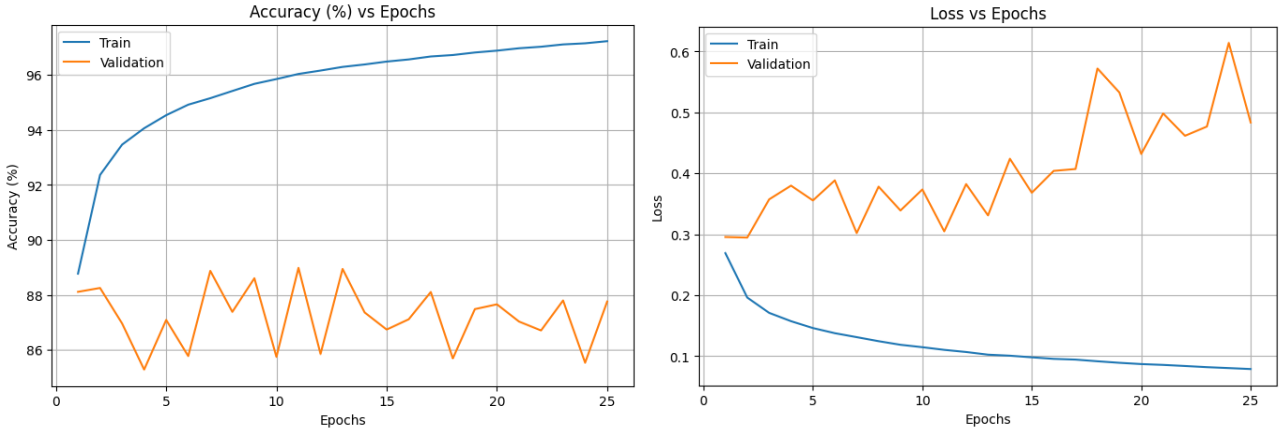
(b) Losses

Figure 18: Learning Rate $1e-2$



(a) Accuracies (b) Losses

Figure 19: learning Rate $1e-4$



(a) Accuracies (b) Losses

Figure 20: Learning Rate $1e-5$

A learning rate of $1e-2$ is not effective for training VGG-16. The optimal learning rates for VGG-16 are $1e-4$ and $1e-5$.

Using higher learning rates, such as $1e-2$ and $1e-3$, destabilizes the training process, causing the model to diverge. This instability leads to poor convergence, with the model predicting all zeros from the initial epochs. The likely reason is that large weight updates prevent the network from learning meaningful features, pushing it into a state where activations become ineffective.

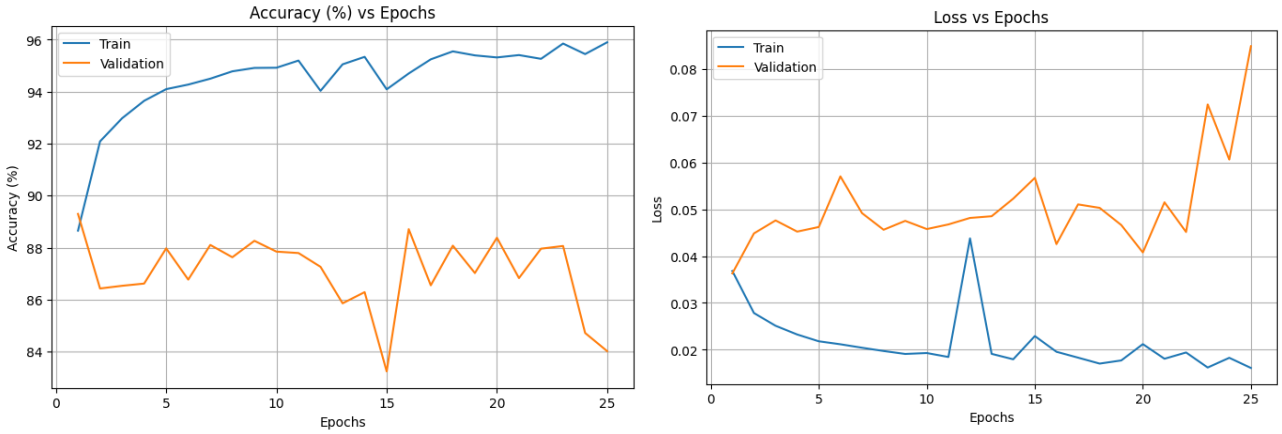
Thus, to ensure stable training and proper feature learning, we use $1e-4$ or $1e-5$ as the learning rate for VGG-16.

Set	Metric	Learning Rate $1e-4$			Learning Rate $1e-5$		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	95.37	96.15	95.63	95.85	96.89	97.23
	Precision	0.96	0.97	0.96	0.96	0.97	0.98
	Recall	0.95	0.95	0.95	0.95	0.96	0.97
	F1 Score	0.95	0.96	0.96	0.96	0.97	0.97
Validation	Accuracy (%)	88.31	85.45	87.28	85.74	87.66	87.76
	Precision	0.96	0.94	0.96	0.96	0.96	0.96
	Recall	0.80	0.76	0.78	0.75	0.79	0.79
	F1 Score	0.87	0.84	0.86	0.84	0.86	0.87
Test	Accuracy (%)	86.655	84.430	86.365	84.308	85.147	85.529
	Precision	0.962	0.954	0.967	0.959	0.956	0.964
	Recall	0.763	0.723	0.753	0.717	0.737	0.738
	F1 Score	0.851	0.823	0.847	0.820	0.832	0.836
Max Accuracy on Test Set (%)		**89.8712**			**88.6078**		

Table 11: Comparison Between Learning Rates for VGG16

3.4 Loss Functions

3.4.1 Focal Loss



(a) Accuracies

(b) Losses

Figure 21: VGG16 with Focal Loss

Using the same logic as in ResNet, applying focal loss does not significantly impact the performance of VGG-16.

Focal loss is designed to address class imbalance by down-weighting easy examples and focusing more on hard-to-classify samples. However, since the dataset is well-balanced, focal loss behaves similarly to standard cross-entropy loss, leading to no noticeable improvement in model performance.

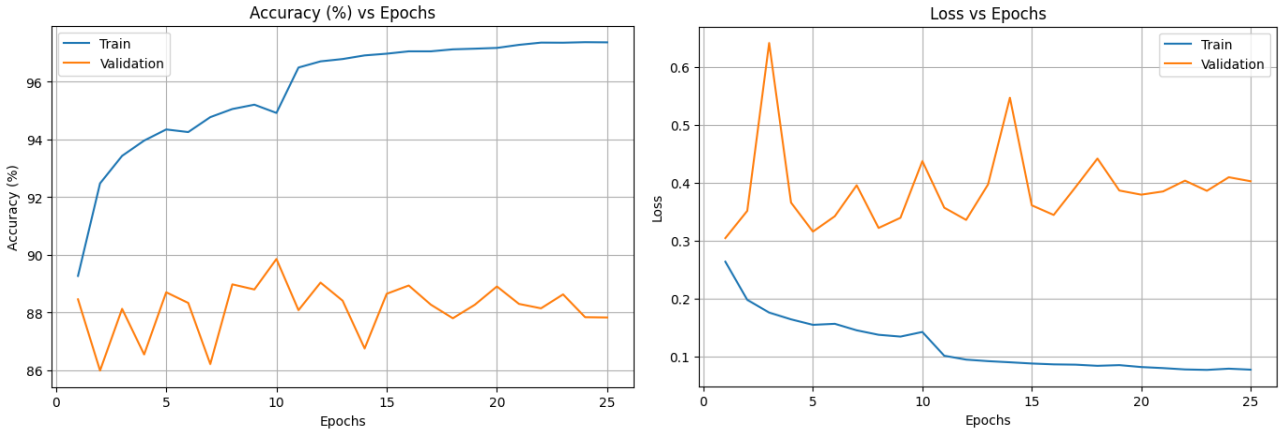
Thus, in this case, focal loss does not provide any advantage over cross-entropy loss.

Set	Metric	Cross-Entropy Loss			Focal Loss		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	95.37	96.15	95.63	94.92	95.32	95.90
	Precision	0.96	0.97	0.96	0.96	0.96	0.97
	Recall	0.95	0.95	0.95	0.94	0.94	0.95
	F1 Score	0.95	0.96	0.96	0.95	0.95	0.96
Validation	Accuracy (%)	88.31	85.45	87.28	87.84	88.38	84.01
	Precision	0.96	0.94	0.96	0.96	0.97	0.98
	Recall	0.80	0.76	0.78	0.79	0.80	0.69
	F1 Score	0.87	0.84	0.86	0.87	0.87	0.81
Test	Accuracy (%)	86.655	84.430	86.365	88.117	85.666	81.833
	Precision	0.962	0.954	0.967	0.969	0.970	0.984
	Recall	0.763	0.723	0.753	0.788	0.736	0.647
	F1 Score	0.851	0.823	0.847	0.869	0.837	0.781
Max Accuracy on Test Set (%)		**89.8712**			**88.1165**		

Table 12: Comparison between Cross-Entropy and Focal Loss for VGG16

3.5 Learning Rate Scheduling

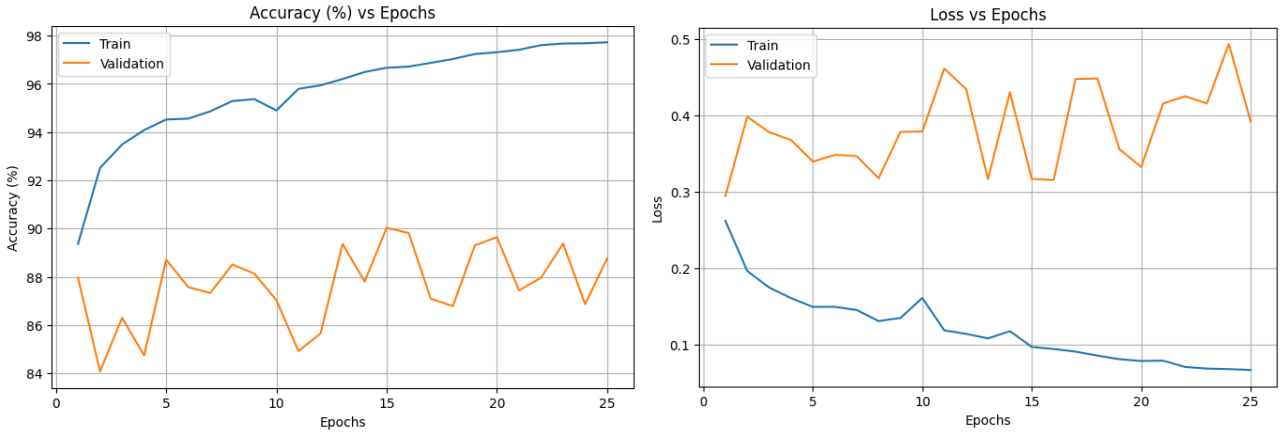
3.5.1 StepLR Scheduling



(a) Accuracies (b) Losses

Figure 22: StepLR Scheduling for VGG16

3.5.2 Cosine Annealing LR Scheduling



(a) Accuracies (b) Losses

Figure 23: Cosine Annealing Scheduling for VGG16

Set	Metric	StepLR 10	StepLR 20	StepLR 25	Cosine LR 10	Cosine LR 20	Cosine LR 25
Train	Accuracy (%)	94.92	97.17	97.37	94.90	97.31	97.73
	Precision	0.96	0.98	0.98	0.96	0.98	0.98
	Recall	0.94	0.97	0.97	0.94	0.97	0.97
	F1 Score	0.95	0.97	0.97	0.95	0.97	0.98
Validation	Accuracy (%)	89.86	88.91	87.83	87.03	89.64	88.76
	Precision	0.94	0.96	0.97	0.96	0.96	0.96
	Recall	0.85	0.81	0.78	0.77	0.82	0.80
	F1 Score	0.89	0.88	0.87	0.86	0.89	0.88
Test	Accuracy (%)	88.742	87.924	86.883	85.275	87.506	87.756
	Precision	0.945	0.967	0.974	0.967	0.967	0.969
	Recall	0.822	0.785	0.758	0.731	0.776	0.780
	F1 Score	0.879	0.867	0.852	0.832	0.861	0.864
Max Accuracy on Test Set (%)		**88.7421**			**89.9231**		

Table 13: Comparison between different Scheduling Techniques for VGG16

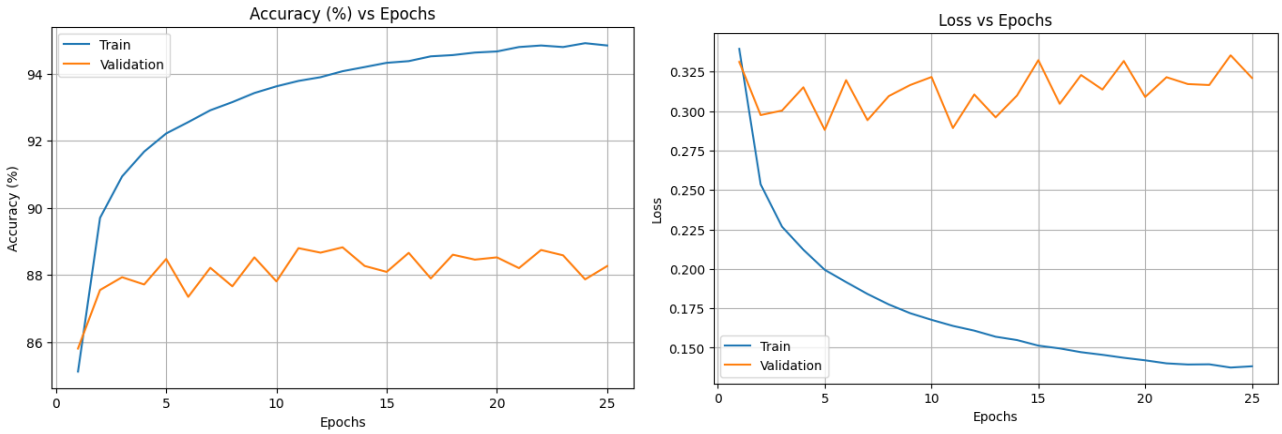
Similar to ResNet-18, Cosine Annealing works well with VGG-16.

The gradual reduction in the learning rate helps stabilize training, ensuring smooth convergence and reducing fluctuations in loss and accuracy. This scheduling strategy allows the model to make finer updates in later epochs, leading to better generalization.

Due to its effectiveness in stabilizing training and improving convergence, we choose Cosine Annealing for VGG-16 as well.

3.6 Optimizers

3.6.1 SGD optimizer



(a) Accuracies

(b) Losses

Figure 24: VGG16 with Stochastic Gradient Descent Optimizer

Set	Metric	Adam 10	Adam 20	Adam 25	SGD 10	SGD 20	SGD 25
Train	Accuracy (%)	94.90	97.31	97.73	93.62	94.66	94.84
	Precision	0.96	0.98	0.98	0.94	0.95	0.96
	Recall	0.94	0.97	0.97	0.93	0.94	0.94
	F1 Score	0.95	0.97	0.98	0.94	0.95	0.95
Validation	Accuracy (%)	87.03	89.64	88.76	87.81	88.52	88.26
	Precision	0.96	0.96	0.96	0.94	0.92	0.94
	Recall	0.77	0.82	0.80	0.81	0.84	0.82
	F1 Score	0.86	0.89	0.88	0.87	0.88	0.87
Test	Accuracy (%)	85.275	87.506	87.756	86.722	87.155	86.136
	Precision	0.967	0.967	0.969	0.948	0.929	0.948
	Recall	0.731	0.776	0.780	0.777	0.804	0.765
	F1 Score	0.832	0.861	0.864	0.854	0.862	0.847
Max Accuracy on Test Set (%)		**89.9231**			**88.2904**		

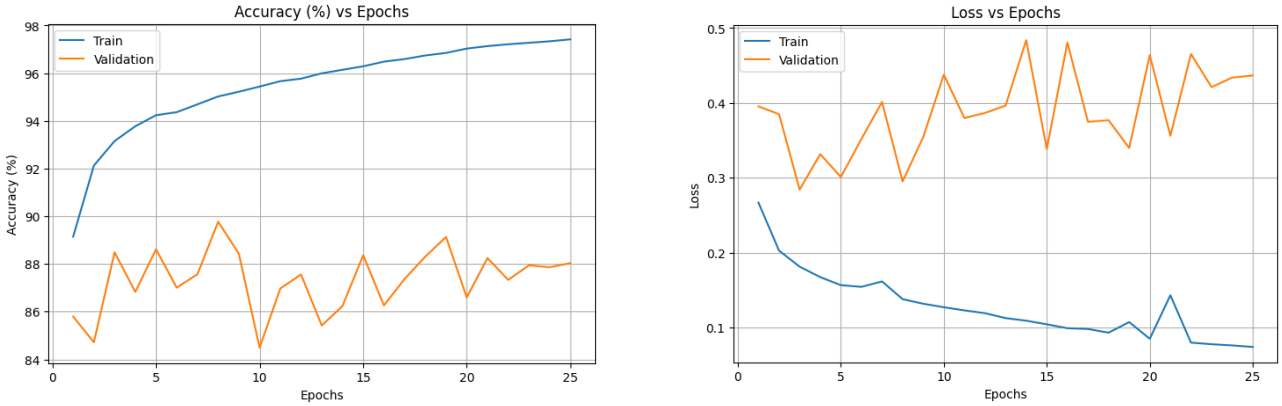
Table 14: Comparison between different optimizers for VGG16

The SGD optimizer results in significantly fewer fluctuations in VGG-16 training while maintaining accuracy. Unlike adaptive optimizers such as Adam, which may introduce instability in training, SGD provides smoother convergence by applying consistent weight updates.

Since this stability is achieved without a significant loss in accuracy, we choose SGD as the optimizer for further experiments.

3.7 Number of Layers

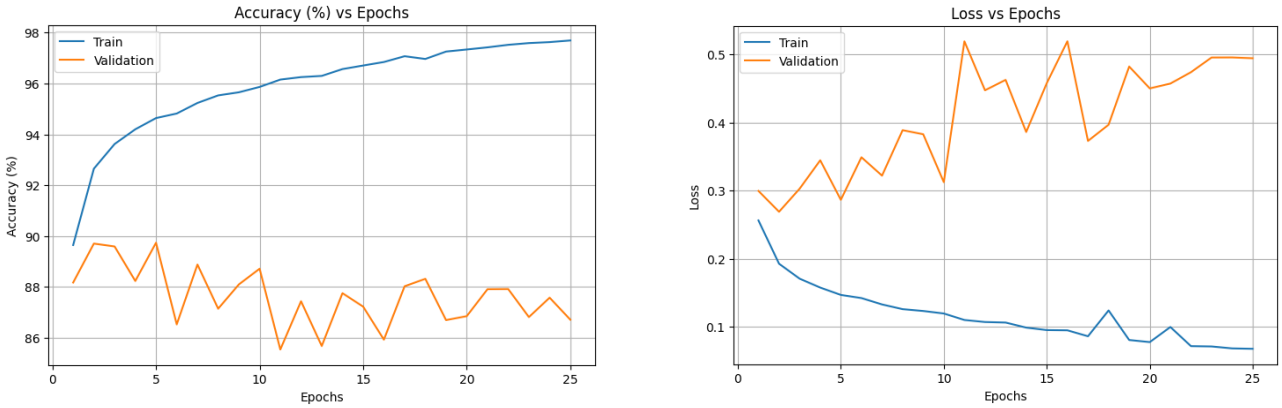
3.7.1 VGG11



(a) Accuracies (b) Losses

Figure 25: VGG11

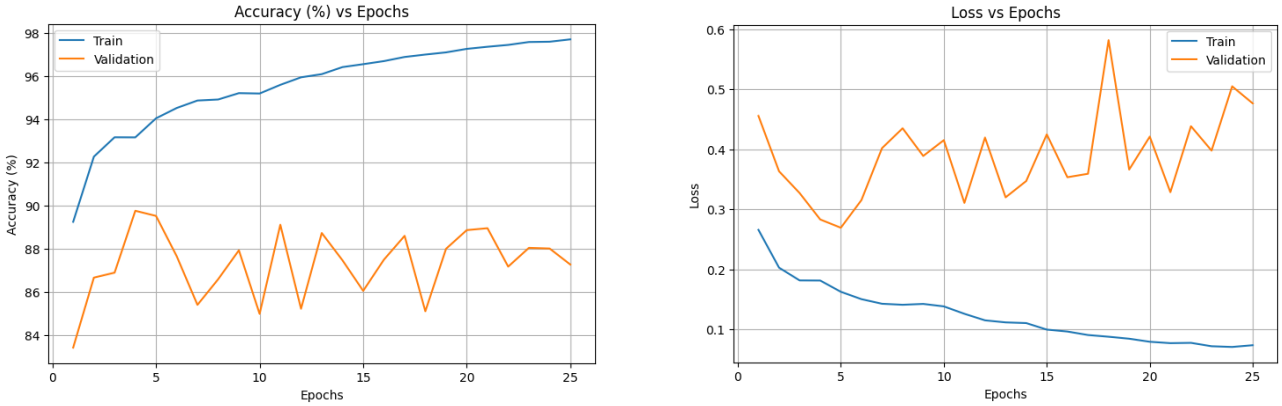
3.7.2 VGG13



(a) Accuracies (b) Losses

Figure 26: VGG13

3.7.3 VGG19



(a) Accuracies

(b) Losses

Figure 27: VGG19

Set	Metric	VGG11			VGG13		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	95.44	97.03	97.42	95.86	97.34	97.69
	Precision	0.96	0.98	0.98	0.97	0.98	0.98
	Recall	0.95	0.96	0.97	0.95	0.97	0.97
	F1 Score	0.95	0.97	0.97	0.96	0.97	0.98
Validation	Accuracy (%)	84.49	86.60	88.03	88.71	86.85	86.71
	Precision	0.96	0.96	0.96	0.95	0.97	0.97
	Recall	0.72	0.76	0.79	0.82	0.76	0.76
	F1 Score	0.82	0.85	0.87	0.88	0.85	0.85
Test	Accuracy (%)	81.656	84.686	85.388	85.718	86.429	85.422
	Precision	0.964	0.968	0.971	0.956	0.976	0.978
	Recall	0.657	0.717	0.730	0.749	0.747	0.725
	F1 Score	0.782	0.824	0.833	0.840	0.846	0.833
Max Accuracy on Test Set (%)		**88.205**			**88.2782**		

Table 15: Comparison Between Number of Layers in VGG

Set	Metric	VGG16			VGG19		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	94.90	97.31	97.73	95.20	97.27	97.72
	Precision	0.96	0.98	0.98	0.96	0.98	0.98
	Recall	0.94	0.97	0.97	0.94	0.97	0.97
	F1 Score	0.95	0.97	0.98	0.95	0.97	0.98
Validation	Accuracy (%)	87.03	89.64	88.76	84.97	88.86	87.26
	Precision	0.96	0.96	0.96	0.96	0.96	0.97
	Recall	0.77	0.82	0.80	0.73	0.81	0.77
	F1 Score	0.86	0.89	0.88	0.83	0.88	0.86
Test	Accuracy (%)	85.275	87.506	87.756	86.334	88.406	87.061
	Precision	0.967	0.967	0.969	0.969	0.970	0.977
	Recall	0.731	0.776	0.780	0.751	0.793	0.759
	F1 Score	0.832	0.861	0.864	0.846	0.872	0.854
Max Accuracy on Test Set (%)		**89.9231**			**89.5172**		

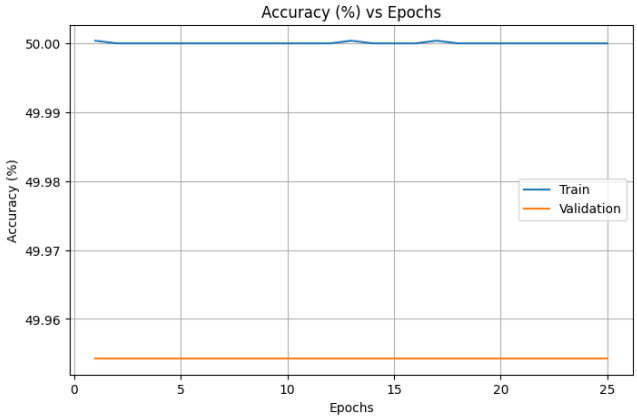
Table 16: Comparison Between Number of Layers in VGG

All four VGG variants perform approximately the same in terms of accuracy and generalization. Despite differences in depth and architecture, their overall performance remains similar, likely because the dataset size and complexity do not fully exploit the advantages of deeper VGG models.

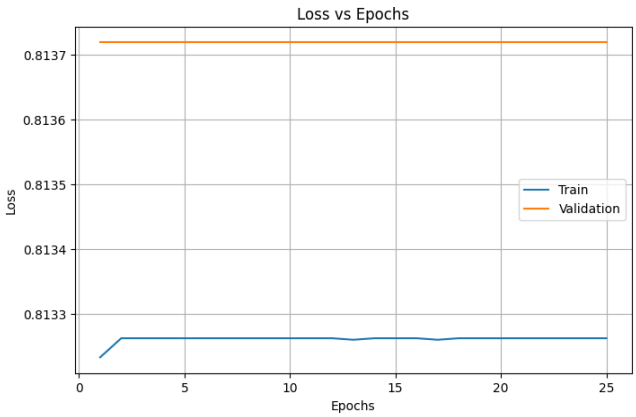
Thus, any of the VGG models can be used without a significant difference in results.

4 Ablation studies on Custom Architecture

4.1 Learning Rate

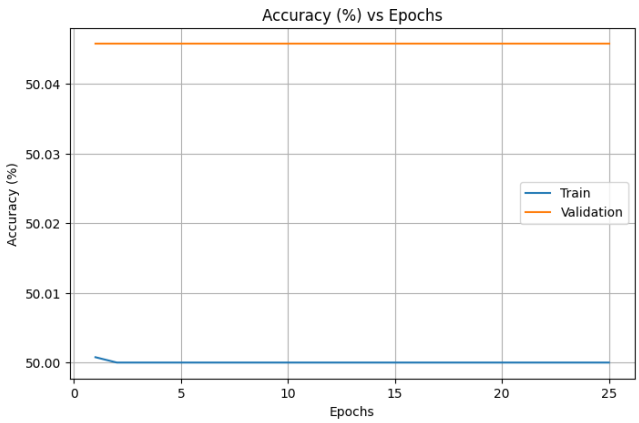


(a) Accuracies

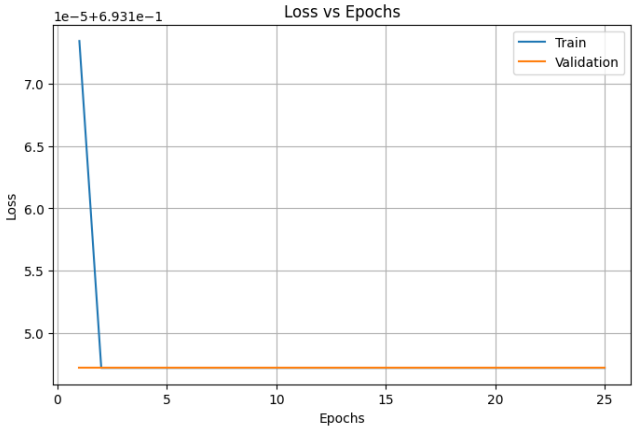


(b) Losses

Figure 28: Learning Rate $1e-2$

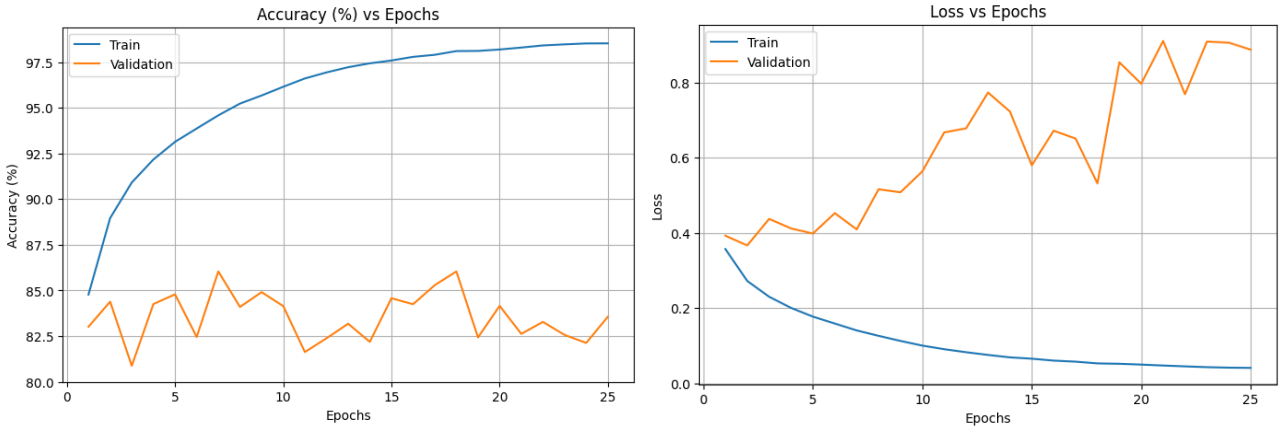


(a) Accuracies



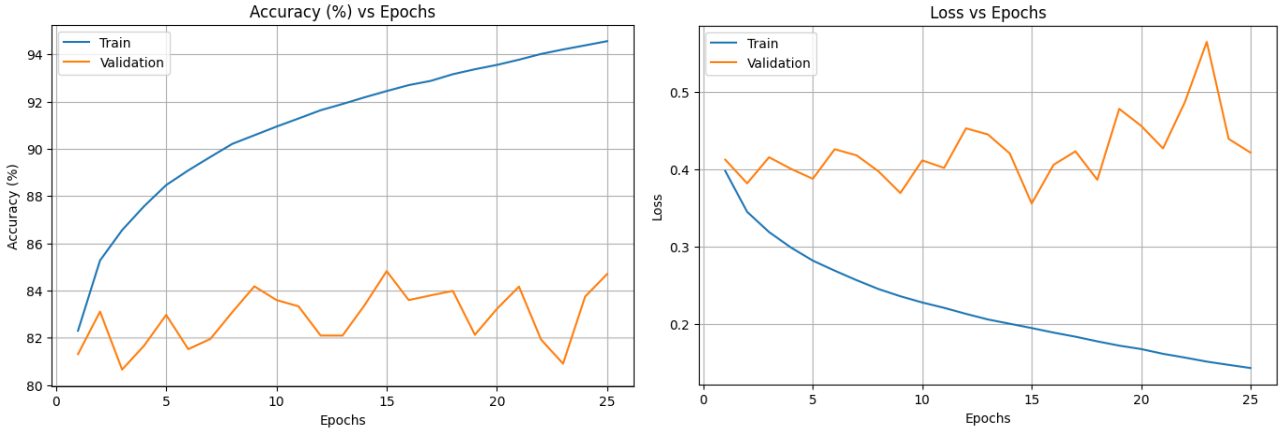
(b) Losses

Figure 29: Learning Rate $1e-3$



(a) Accuracies (b) Losses

Figure 30: Learning Rate $1e-4$



(a) Accuracies (b) Losses

Figure 31: Learning Rate $1e-5$

Similar to VGG-16, learning rates of $1e-4$ and $1e-5$ work well for the custom architecture.

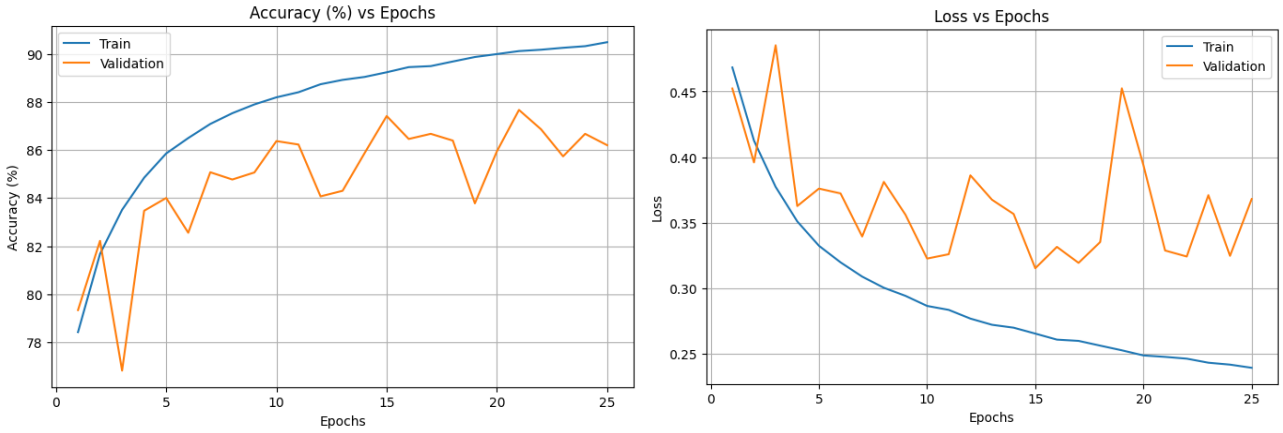
Higher learning rates, such as $1e-3$ or $1e-2$, destabilize training, leading to poor convergence and unreliable predictions. In contrast, lower learning rates ensure smoother optimization, allowing the model to learn meaningful features effectively.

Thus, for stable training and optimal performance, we select $1e-4$ and $1e-5$ as the preferred learning rates for the custom architecture.

Set	Metric	Learning Rate $1e-4$			Learning Rate $1e-3$		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	96.15	98.19	98.53	90.94	93.55	94.56
	Precision	0.97	0.98	0.99	0.91	0.94	0.95
	Recall	0.96	0.98	0.98	0.91	0.93	0.94
	F1 Score	0.96	0.98	0.99	0.91	0.94	0.95
Validation	Accuracy (%)	84.14	84.16	83.56	83.60	83.22	84.70
	Precision	0.92	0.93	0.91	0.91	0.92	0.90
	Recall	0.75	0.74	0.75	0.75	0.73	0.78
	F1 Score	0.83	0.82	0.82	0.82	0.81	0.84
Test	Accuracy (%)	80.350	79.922	80.096	78.336	78.729	79.309
	Precision	0.913	0.916	0.905	0.896	0.912	0.890
	Recall	0.671	0.658	0.672	0.641	0.636	0.669
	F1 Score	0.773	0.766	0.771	0.747	0.749	0.764
Max Accuracy on Test Set (%)		**84.3994**			**81.3202**		

Table 17: Comparison Between Learning Rates for Custom Architecture

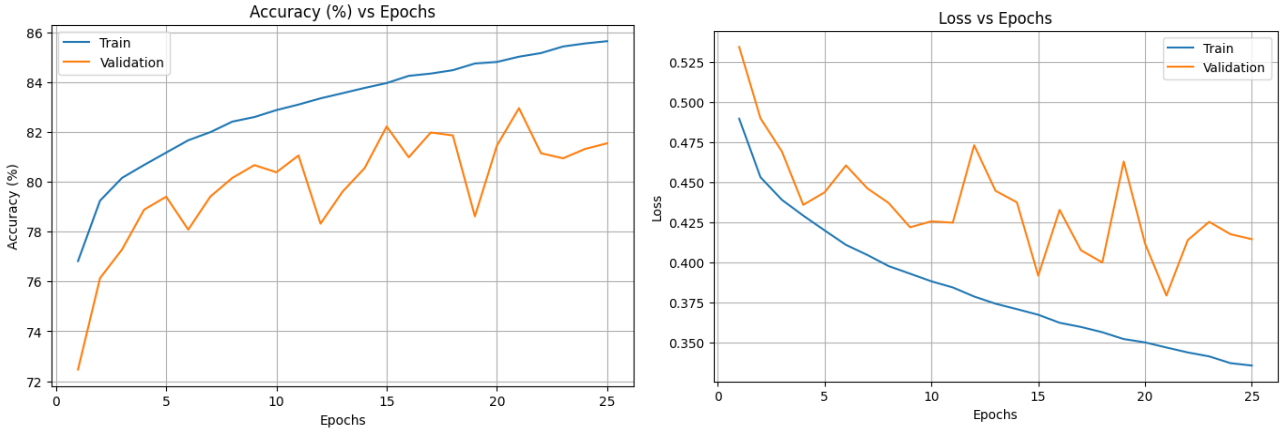
4.2 Augmentation Techniques



(a) Accuracies

(b) Losses

Figure 32: Custom Architecture with Data Augmentations on Learning Rate $1e-4$



(a) Accuracies

(b) Losses

Figure 33: Custom Architecture with Data Augmentations on Learning Rate $1e-5$

Set	Metric	Augm. with LR $1e-4$ 10	Augm. with LR $1e-4$ 20	Augm. with LR $1e-4$ 25	Augm. with LR $1e-5$ 10	Augm. with LR $1e-5$ 20	Augm. with LR $1e-5$ 25
Train	Accuracy (%)	88.19	89.99	90.49	82.88	84.81	85.64
	Precision	0.89	0.91	0.92	0.83	0.85	0.86
	Recall	0.87	0.89	0.89	0.83	0.85	0.85
	F1 Score	0.88	0.90	0.90	0.83	0.85	0.86
Validation	Accuracy (%)	86.37	85.95	86.20	80.38	81.45	81.54
	Precision	0.87	0.91	0.92	0.82	0.88	0.87
	Recall	0.85	0.80	0.80	0.78	0.72	0.74
	F1 Score	0.86	0.85	0.85	0.80	0.80	0.80
Test	Accuracy (%)	82.712	83.563	83.646	78.275	78.632	78.824
	Precision	0.879	0.922	0.926	0.820	0.884	0.872
	Recall	0.758	0.733	0.731	0.725	0.659	0.676
	F1 Score	0.814	0.817	0.817	0.769	0.755	0.761
Max Accuracy on Test Set (%)		**86.1816**			**80.246**		

Table 18: Comparison of Custom Arch. with Data Augmentation

Similar to the previous models, applying data augmentation improves accuracy by approximately 2%.

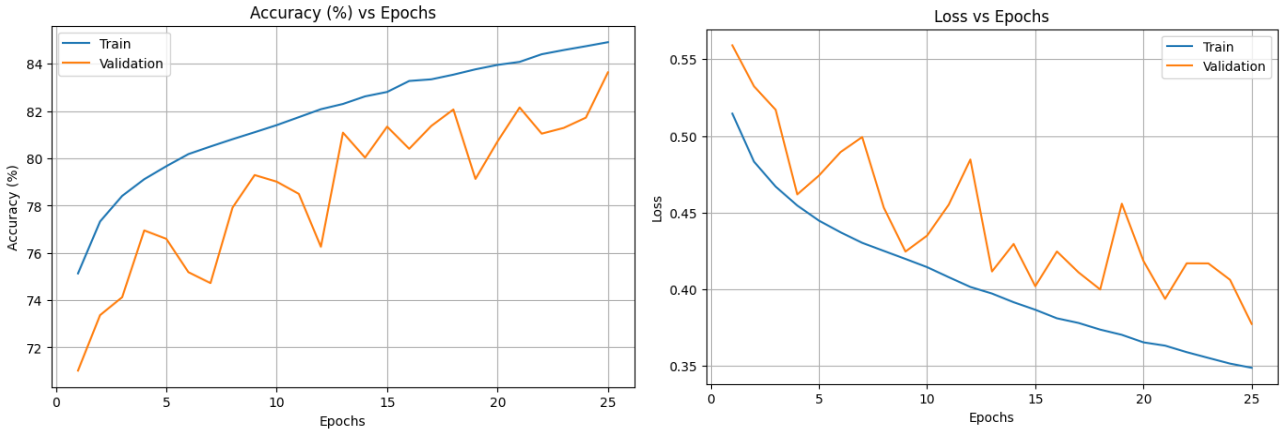
Augmentation introduces variations in the training data, helping the model generalize better to unseen samples. By preventing overfitting and encouraging the learning of more robust features, it leads to improved performance on the test set.

Thus, data augmentation is beneficial for enhancing the model's accuracy and generalization.

4.3 Optimizers

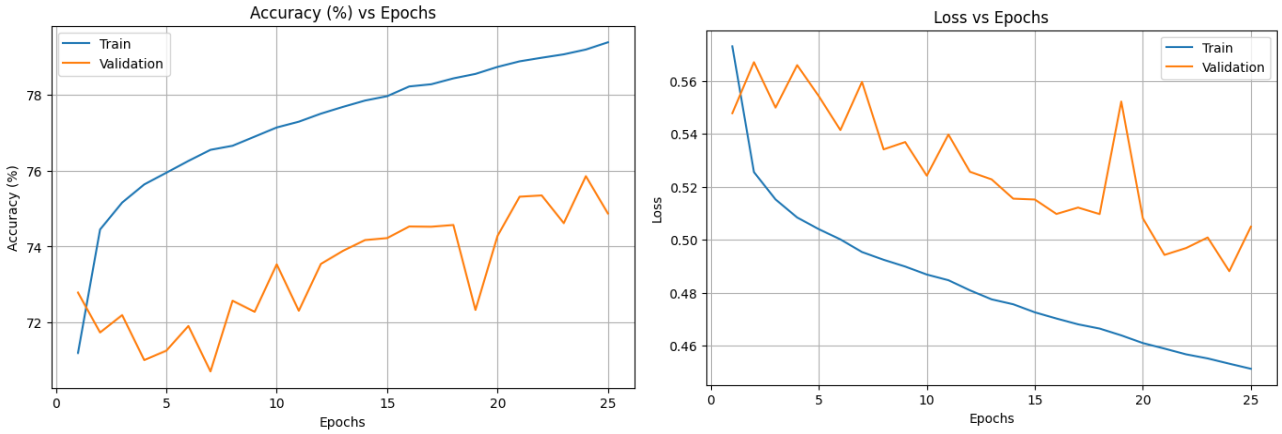
4.3.1 Adam

4.3.2 SGD



(a) Accuracies (b) Losses

Figure 34: SGD Optimizer with Learning Rate $1e-4$



(a) Accuracies (b) Losses

Figure 35: SGD Optimizer with Learning Rate $1e-5$

SGD struggles with convergence in the custom architecture compared to Adam.

Due to its adaptive learning rate mechanism, Adam optimizes weight updates more effectively, leading to faster and more stable convergence. In contrast, SGD requires careful tuning of the learning rate and momentum to achieve similar results, making it less efficient for this architecture.

Thus, we choose Adam as the preferred optimizer for the custom architecture.

Set	Metric	SGD with LR $1e-4$			SGD with LR $1e-5$		
		10	**20**	**25**	**10**	**20**	**25**
Train	Accuracy (%)	81.41	83.96	84.92	77.13	78.73	79.38
	Precision	0.81	0.84	0.85	0.77	0.78	0.79
	Recall	0.82	0.84	0.85	0.78	0.80	0.80
	F1 Score	0.81	0.84	0.85	0.77	0.79	0.80
Validation	Accuracy (%)	79.01	80.72	83.64	73.53	74.28	74.87
	Precision	0.80	0.88	0.83	0.80	0.80	0.78
	Recall	0.78	0.71	0.85	0.62	0.64	0.69
	F1 Score	0.79	0.79	0.84	0.70	0.71	0.73
Test	Accuracy (%)	77.200	76.355	79.376	70.456	72.400	73.886
	Precision	0.797	0.876	0.823	0.788	0.796	0.786
	Recall	0.730	0.613	0.748	0.560	0.602	0.656
	F1 Score	0.762	0.722	0.784	0.654	0.685	0.715
Max Accuracy on Test Set (%)		**79.3762**			**75.6561**		

Table 19: Comparison of Custom Architecture for Stochastic Gradient Descent Optimizer

5 Ablation Studies Conclusions

The best-performing model is VGG-16 when trained with the Adam optimizer and Cosine Annealing learning rate scheduling. This combination provides stable convergence, reduced fluctuations, and improved generalization, leading to optimal performance.

6 Competitive Model Improvement

To further enhance model performance, the following improvements were implemented:

1. Added Batch Normalization and Dropout in the VGG model, which improved convergence and reduced overfitting.
2. Enhanced the custom architecture by incorporating additional Inception blocks and residual layers for better feature extraction.
3. Implemented an ensemble of the best-performing VGG, custom, and ResNet models to generate predictions for the test dataset.

With these improvements, the final testing accuracy reached 91%.