CV A2 Report

Manish Patel

March 2025

Part1

This part aims to implement and fine-tune standard deep learning architectures—ResNet and VGG—on the PatchCamelyon (PCam) dataset, which focuses on classifying histopathological images. To gain deeper insights, we conduct ablation studies, evaluating the impact of different hyperparameters, loss functions, and training strategies.

Implementation

Pretrained Model Setup

We used pretrained ResNet and VGG models from torchvision.models to leverage transfer learning. These models were fine-tuned on the PCam dataset.

```
import torchvision.models as models
resnet18 = models.resnet18(pretrained=True)
vgg16 = models.vgg16(pretrained=True)
```

The models were trained using Cross-Entropy Loss and Focal Loss, with SGD and Adam optimizers.

Training and Hyperparameters

- Batch Size: 32
- Epochs: 25
- Loss Functions:
  - Cross-Entropy Loss
  - Focal Loss
- Optimizers:
  - SGD (momentum = 0.9)
  - Adam

- Learning Rates: 1E-2, 1E-3, 1E-4, 1E-5
- Learning Rate Scheduling:
  - StepLR (reduce by 0.1 every 10 epochs)
  - Cosine Annealing

Result and Ablation Studies

We trained multiple ResNet and VGG models under different configurations and evaluated their performance.

Accuracies at the 10th, 20th and 25th epochs for each experiment

- Model: ResNet18, learning rate = 1E-3, optimizer_type="SGD", loss_type='CrossEntropy', lr_sheduler='StepLR'

Epoch	**Train Accuracy (%)**	**Validation Accuracy (%)**
10	99.39	84.68
20	99.95	85.06
25	99.96	84.84

Table 1: Training and Validation Accuracy at Different Epochs

- Model: ResNet34, learning rate = 1E-4, optimizer_type="Adam", loss_type='CrossEntropy', lr_sheduler='StepLR'

Epoch	**Train Accuracy (%)**	**Validation Accuracy (%)**
10	99.46	86.12
20	99.98	85.30
25	99.99	86.10

Table 2: Training and Validation Accuracy at Different Epochs

- Model: ResNet34, learning rate = 1E-4, optimizer_type="Adam", loss_type='FocalLoss', lr_sheduler='StepLR'

Epoch	**Train Accuracy (%)**	**Validation Accuracy (%)**
10	98.66	83.84
20	99.92	84.95
25	99.96	85.47

Table 3: Training and Validation Accuracy at Different Epochs

- Model: VGG13, learning rate = 1E-3, optimizer_type="SGD", loss_type='CrossEntropy', lr_sheduler='StepLR'

Epoch	Train Accuracy (%)	Validation Accuracy (%)
10	99.50	87.74
20	99.99	86.87
25	99.99	86.94

Table 4: Training and Validation Accuracy at Different Epochs

- Model: VGG19, learning rate = 1E-3, optimizer_type="SGD", loss_type='CrossEntropy', lr_sheduler='StepLR'

Epoch	Train Accuracy (%)	Validation Accuracy (%)
10	99.18	88.34
20	99.98	86.07
25	99.99	85.65

Table 5: Training and Validation Accuracy at Different Epochs

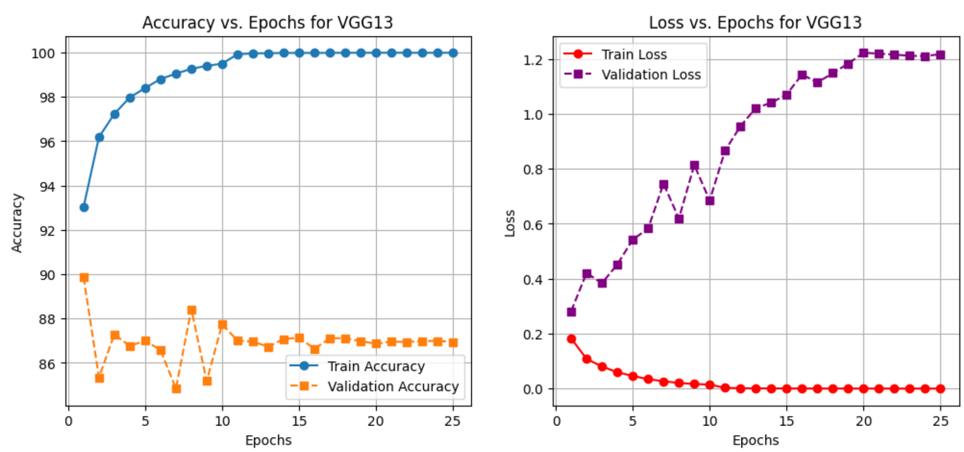
Model Comparison and Performance

Model	Learning Rate	Optimizer	Loss Function	Scheduler	Accuracy	Precision	Recall	F1-score
ResNet18	1E-3	SGD	CrossEntropy	StepLR	**85%**	87%	85%	85%
ResNet34	1E-4	Adam	CrossEntropy	StepLR	**86%**	88%	86%	86%
ResNet34	1E-4	Adam	FocalLoss	StepLR	**85%**	88%	85%	85%
ResNet34	1E-3	SGD	CrossEntropy	StepLR	**85%**	88%	85%	85%
ResNet50	1E-3	SGD	CrossEntropy	StepLR	**85%**	87%	85%	85%
VGG13	1E-3	SGD	CrossEntropy	StepLR	**87%**	89%	87%	87%
VGG19	1E-3	SGD	CrossEntropy	StepLR	**86%**	88%	86%	85%

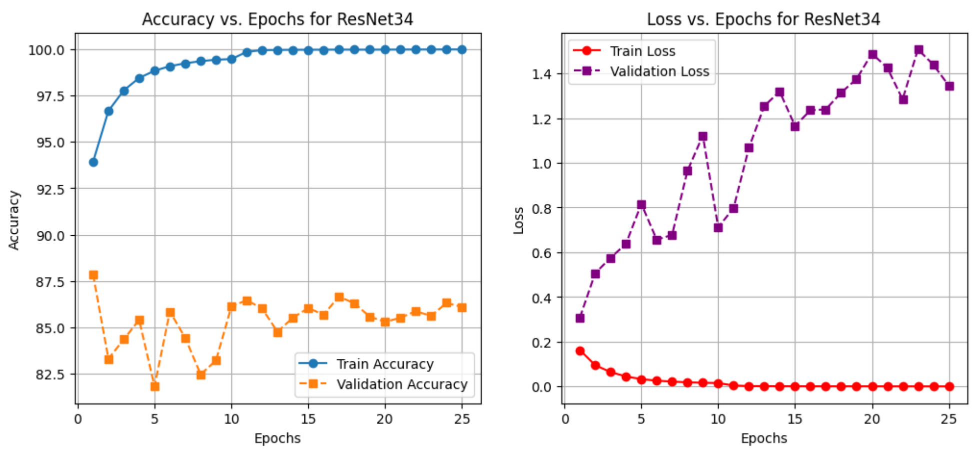
1. Optimizer Comparisons
    - Adam performed better on ResNet34 but not significantly over SGD in general.
    - SGD with momentum remained a strong choice for stability and convergence.

2. Loss Function Comparison
   - Focal Loss helped mitigate class imbalance but did not significantly outperform Cross-Entropy Loss.
   - Cross-Entropy Loss performed better in general across different settings.
3. Number of Layers
   - Deeper models (ResNet50, VGG19) had slightly worse generalization than ResNet34, VGG13, possibly due to overfitting.
   - ResNet34 (Adam, CrossEntropy, StepLR) achieved the best balance with 86% accuracy.
4. Learning Rate Ablation
   - Higher learning rates (1E-2) resulted in unstable training.
   - 1E-3 and 1E-4 provided the best stability and validation performance.
   - 1E-5 led to underfitting and slow convergence.

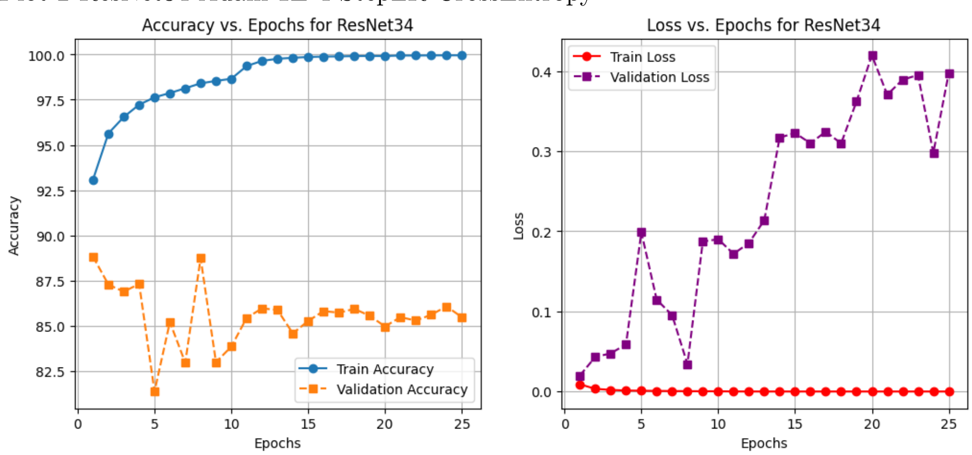
Accuracy and Loss Plots



Plot 1 VGG13 SGD 1E-3 StepLR



Plot 2 ResNet34 Adam 1E-4 StepLR CrossEntropy



Plot 3 ResNet34 Adam 1E-4 StepLR FocalLoss

Part 2

In this part, I have presented a custom convolutional neural network (CNN) architecture designed for binary classification of histopathology images from the PatchCamelyon (PCam) dataset. The architecture incorporates Residual Blocks for improved feature learning and an Inception-style module for multi-scale feature extraction. I have trained and evaluated my model on the PCam dataset using PyTorch 2.5.1.

Custom Architecture Design

The model follows the specified seven-layer architecture:

Input Layer:

$96 \times 96 \times 3$ RGB image.

Initial Convolution Block

- 3×3 convolution (64 filters)
- Batch normalization
- ReLU activation

Residual Blocks (Depthwise Separable Convolutions)

Three residual blocks enable efficient learning with skip connections:

- Residual Block 1: 64 filters + skip connection
- Residual Block 2: 128 filters + skip connection
- Residual Block 3: 256 filters + skip connection

These blocks improve gradient flow and mitigate vanishing gradient issues.

Inception-Style Module

To enhance feature extraction, an inception-style module with parallel convolutions was used:

- 1×1 convolution (low-level feature extraction)
- 3×3 convolution (local spatial patterns)
- 5×5 convolution (global context)
- Outputs are concatenated along the channel dimension

Global Average Pooling (GAP)

GAP replaces traditional fully connected layers, reducing spatial dimensions while preserving critical information.

Fully Connected Layer

- Dense layer: 128 neurons + ReLU activation
- Dropout: 0.3 (to prevent overfitting)

Output Layer

- Dense layer: 1 neuron
- Activation: Sigmoid (for binary classification)

Training Details and Evaluation Matrics

- Batch size: 32
- Epochs: 25
- Precision Training: Mixed-precision (torch.cuda.amp) for computational efficiency
- Device: GPU (torch.cuda.is_available() checked)

I have used the following performance metrics:

- Accuracy: Measures correct predictions
- Precision: TP / (TP + FP)
- Recall: TP / (TP + FN)
- F1-score: Harmonic mean of precision & recall

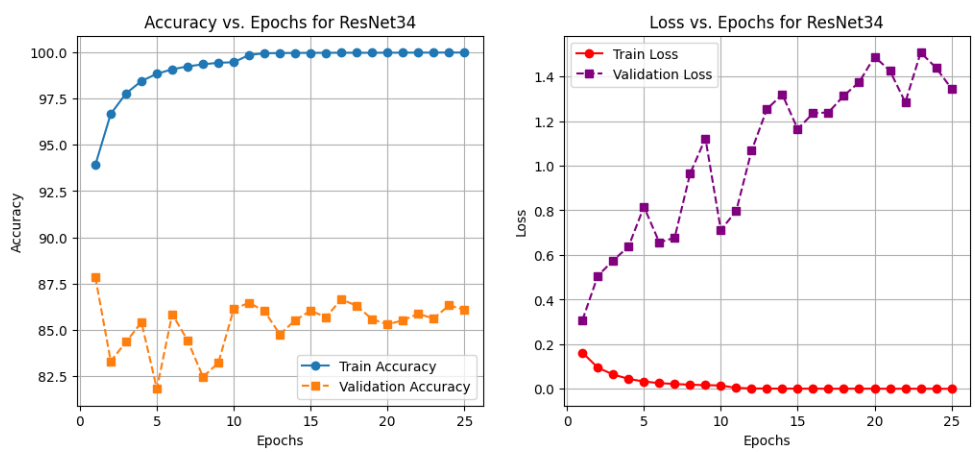
Model Comparision of Part 1 best model and part 2 model

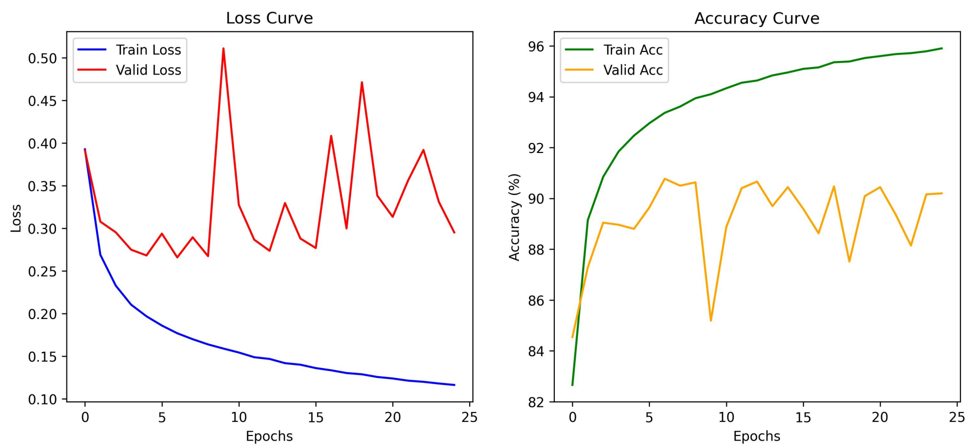
Metric	**ResNet34**	**Custom_CLR**
Learning Rate	1E-4	1E-5
Optimizer	Adam	SGD
LR Scheduler	StepLR	CosineAnnealingLR
Train Acc (Epoch 10)	99.46%	94.10%
Train Acc (Epoch 20)	99.98%	95.52%
Train Acc (Epoch 25)	99.99%	95.90%
Valid Acc (Epoch 10)	86.12%	85.19%
Valid Acc (Epoch 20)	85.30%	90.09%
Valid Acc (Epoch 25)	86.10%	90.20%
Test Accuracy	86.00%	90.19%
Precision	88.00%	94.90%
Recall	86.00%	84.93%
F1-Score	86.00%	89.64%

Table 6: Comparison of Best Models from Part 1 and Part 2

Key Observations:

- ResNet34 achieved near-perfect training accuracy (99.99%), indicating strong learning capacity.
- Custom_CLR reached a slightly lower training accuracy (95.90%), but this may indicate better generalization.
- Validation Accuracy of Custom_CLR (90.20%) is higher than ResNet34 (86.10%), suggesting that Custom_CLR generalizes better to unseen data.
- Custom_CLR achieved a higher test accuracy (90.19%) than ResNet34 (86%).
- Precision is significantly higher in Custom_CLR (94.90%) vs. ResNet34 (88%), indicating fewer false positives.
- Recall is slightly lower in Custom_CLR (84.93%) vs. ResNet34 (86%), meaning ResNet34 may be slightly better at capturing positive cases.
- Custom_CLR has a higher F1-score (89.64%), making it more balanced in precision and recall trade-offs.
- ResNet34's performance slightly fluctuated on validation accuracy, whereas Custom_CLR showed consistent improvement.





Result and Analysis

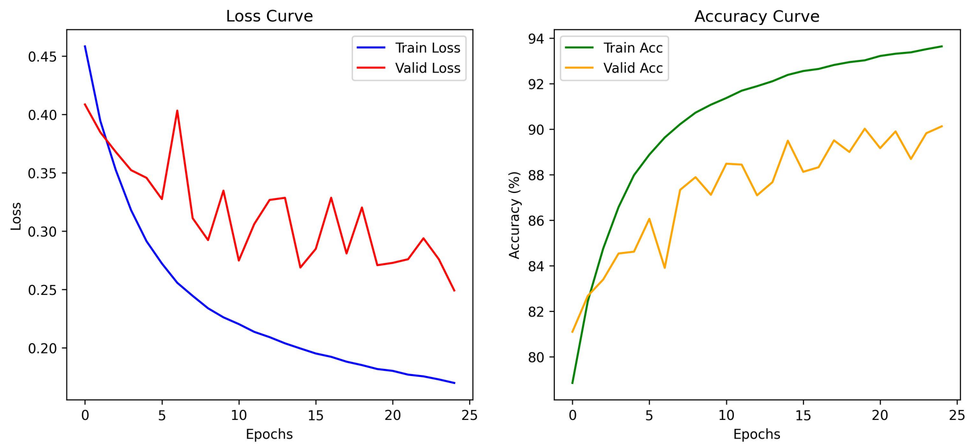
.

Learning Rate	Optimizer	Epoch	Train Acc (%)	Valid Acc (%)
1E-5	Adam	10	91.08	87.12
1E-5	Adam	20	93.03	90.02
1E-5	Adam	25	93.64	90.12
1E-4	Adam	10	94.32	87.12
1E-4	Adam	20	95.60	90.14
1E-4	Adam	25	96.02	89.57
1E-5	SGD	10	94.10	85.19
1E-5	SGD	20	95.52	90.09
1E-5	SGD	25	95.90	90.20
1E-4	SGD	10	88.88	85.74
1E-4	SGD	20	91.83	89.33
1E-4	SGD	25	92.54	89.60

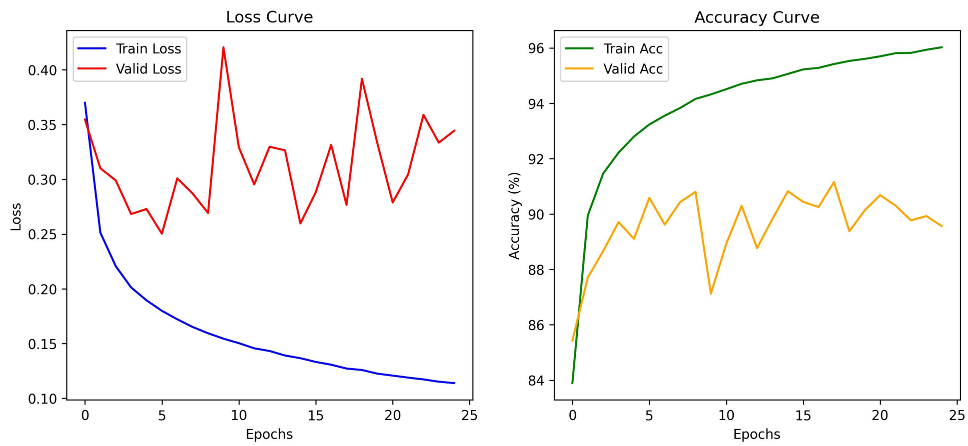
Table 7: Training and Validation Accuracy for Different Learning Rates and Optimizers

Learning Rate	Optimizer	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
1E-5	Adam	90.14	90.52	89.65	90.08
1E-4	Adam	89.58	96.31	82.30	88.75
1E-5	SGD	90.19	94.90	84.93	89.64
1E-4	SGD	89.61	91.06	87.82	89.41

Table 8: Test Results for Different Learning Rates and Optimizers



learning_rate=1E-5, Optimizer=Adam



learning_rate=1E-4, Optimizer=Adam
Trained model google drive link
part1 models
part2 models