COL780 - Computer Vision
Assignment 2: Deep Learning for Image Classification

Ananya Mathur cs5200416@iitd.ac.in
Rajat Mahaur cs5200534@iitd.ac.in

Submission Deadline: 30 March 2025, 11:59 pm

1 Introduction

In this assignment, you will work with the PatchCamelyon (PCam) dataset to develop deep learning models for medical image classification. The PCam dataset consists of 96x96 pixel RGB images of histopathologic scans of lymph node sections. Your task is to classify each image as either containing metastatic tissue (positive) or not (negative).

Dataset: PatchCamelyon (PCam) - `https://github.com/basveeling/pcam`

2 Part 1: Standard Network Implementation and Ablation Studies (45 marks)

Implement and train two standard network architectures: ResNet and VGG. Conduct ablation studies to understand the impact of various hyperparameters and architectural choices.

2.1 Implementation

1. Use pretrained ResNet and VG models from `torchvision.models` to ensure a consistent starting point for all participants.
2. Fine-tune the models on the PCam dataset.
3. Report the baseline accuracy for both models and compare their performance across different hyperparameters.

Pretrained Model Loading in PyTorch:

```
import torchvision.models as models

resnet18 = models.resnet18(pretrained=True)
vgg16 = models.vgg16(pretrained=True)
```

2.2 Training and Hyperparameters

- **Batch Size:** 32
- **Epochs:** 25
- **Loss Function:**
  - Cross-Entropy Loss
  - Focal Loss
- **Optimizer Choices:**
  - Stochastic Gradient Descent (SGD) with momentum = 0.9
  - Adam Optimizer

2.3 Ablation Studies

To assess the impact of architectural and hyperparameter choices, perform the following experiments:

1. **Learning Rate:** Experiment with values $[1e-2, 1e-3, 1e-4, 1e-5]$.
2. **Number of Layers:**
   (a) For ResNet, experiment with $\{\text{ResNet18}, \text{ResNet34}, \text{ResNet50}\}$.
   (b) For VGG, try $\{\text{VGG11}, \text{VGG13}, \text{VGG16}, \text{VGG19}\}$.
3. **Effect of Skip Connections:** Remove skip connections from ResNet and compare performance.
4. **Loss Function Comparison:** Compare Cross-Entropy Loss and Focal Loss.
5. **Learning Rate Scheduling:** Implement learning rate decay strategies:
   - StepLR: Reduce by 0.1 every 10 epochs.
   - Cosine Annealing.
6. **Data Augmentation Techniques:** Evaluate performance with and without augmentation:
   - Random horizontal/vertical flips
   - Rotation ($\pm 15^\circ$)
   - Color jittering
7. **Optimizer Comparisons:** Compare SGD with momentum and Adam.
8. Plot the accuracy and loss vs epoch curves and report accuracies at the 10^{th}, 20^{th} and 25^{th} epochs.

2.4 Evaluation and Reporting

- Report accuracy, precision, recall, and F1-score for each experiment.
- Provide loss and accuracy plots over epochs.
- Summarize findings in a table to compare the impact of different hyperparameters.

3 Part 2: Custom Architecture (35 marks)

You will design and implement the following given custom architecture from scratch:

3.1 Architecture Specifications

1. **Input Layer:** $96 \times 96 \times 3$ RGB image.
2. **Initial Convolution Block:** 3×3 convolution (64 filters) + batch normalization + ReLU activation.
3. **Residual Blocks:** Three residual blocks with depthwise separable convolutions:
   (a) **Residual Block 1:** 64 filters + skip connection.
   (b) **Residual Block 2:** 128 filters + skip connection.
   (c) **Residual Block 3:** 256 filters + skip connection.
4. **Inception-style Module:** Parallel convolutional layers:
   (a) 1×1 convolution.
   (b) 3×3 convolution.
   (c) 5×5 convolution.

   The outputs are concatenated along the channel dimension.
5. **Global Average Pooling:** Reduces spatial dimensions before classification.
6. **Fully Connected Layer:** Dense layer (128 neurons) + ReLU + dropout (0.3).
7. **Output Layer:** Dense layer with 1 neuron and sigmoid activation for binary classification.

3.2 Implementation and Training

1. Implement the custom architecture using PyTorch.
2. Train the model on the PCam dataset.
3. Report the ablation studies, accuracy and loss curves and compare it with the standard models from Part 1.

4 Part 3: Model Improvement Competition (10 marks)

Improve either the best-performing standard network from Part 1 or the custom architecture from Part 2. The goal is to achieve the highest possible accuracy on the PCam dataset.

4.1 Suggested Modifications

You may apply the following techniques to improve model performance:

1. Data augmentation techniques
2. Regularization methods (e.g., dropout, batch normalization)
3. Ensemble methods
4. Learning rate scheduling
5. Architectural modifications (limited to adding, removing, or resizing layers)

4.2 Implementation and Evaluation

1. Implement your chosen modifications, either from the suggested list above or any other method of your choice.
2. Fine-tune the improved model on the PCam dataset.
3. Report the final accuracy and compare it with your previous results.
4. Provide a brief explanation of your approach and the rationale behind its expected performance improvements.

5 Submission Guidelines

1. Submit a **report (PDF) (10 Marks)** detailing your implementations, experiments, and results for all three parts.
2. Include plots and tables to visualize your results.
3. Provide your source code (containing all .py files) as a separate ZIP file (entry_number.zip).
4. Include a README file with instructions on how to run your code.
5. Submit a script named `prediction_<Partnum>.py` for Part 1 (which loads model with best-performing hyperparameters), Part 2 and Part 3. This script should:
   (a) Take as input a folder (`test_images`) containing test images from PCam.
   (b) Load a pretrained model.
   (c) Generate predictions (positive or negative) for each test image.
   (d) Write the predictions to a text file named `results.txt`.
6. Upload the trained models for each part to a cloud drive and include the shareable link in the report. Only models uploaded before the submission deadline will be considered. During the demonstration, you must download and use the same models for `prediction_<Partnum>.py`.

6 Evaluation Criteria

1. Correctness and completeness of implementations
2. Quality and depth of ablation studies and analysis in the report
3. Creativity and effeciveness of custom architecture
4. Performance improvement in the competitive part

7 Note

- For uniformity, please use PyTorch version 2.5.1 throughout the assignment.
- All assignment related doubts and important updates will be communicated via Piazza.
- For students who have not signed up yet,
  Class link: `https://piazza.com/iitd.ac.in/spring2025/col780`
  Access code: g6sydru509f