CPSC599 Assignment 3 Report

Sam Hoang Hong (UCID: 30088823)
Nam Nguyen Vu (UCID: 30154892)
Tin Le (UCID: 30089727)

Our approach started with splitting the dataset, with 80% for training and 20% for validation. This separation helps evaluate the model's performance on data it has not seen during training, providing a reliable estimate of generalization and checking for overfitting. Some of our methods to avoid overfitting include:

- **Flipping images horizontally**. This helps the model generalize to different object orientations.
- **Rotating images**. This enhances the model's ability to recognize objects from various perspectives.

We also standardized pixel values to have zero mean and unit variance. This normalization helps stabilize the training process. For training, we resized the images to 400x400px and applied random flip and rotation transformations. For validation, we also resized the images to 400x400px, but without any augmentations. This ensures that the validation performance is evaluated on unaltered data.

Our model uses 4 classifiers learned from class: Linear, BatchNorm1d, ReLU, and Dropout. The linear layer helps map the extracted features to the output classes. Batch normalization helps stabilize and accelerate training by normalizing the input to each layer. ReLU introduces non-linearity, enabling the model to learn complex mappings between inputs and outputs. Lastly, to help prevent overfitting, we use dropout to randomly set a fraction of input units to zero during training.

We chose Stochastic Gradient Descent with momentum as the optimizer. Since the dataset is fairly small, with 4000 images in 100 classes, we chose a batch size of 16. Learning rate is set to = 0.01, and momentum is set to = 0.9. CrossEntropyLoss was selected as our criterion, as it’s commonly used for multi-class classification tasks.

An approach we used that significantly improved our model was using the learning rate scheduler **torch.optim.lr_scheduler.ExponentialLR()**. This was also applied in the training step. We found out that applying the scheduler decreases the learning rate exponentially over time. This is beneficial for fine-tuning and stabilizing the learning process. Additionally, applying the learning rate scheduler during testing (using the validation accuracy) is an interesting approach. It adjusts the learning rate dynamically based on the model's performance, potentially fine-tuning it to the specific characteristics of the dataset.

Lastly, for future improvement, we would try to test with different pretrained models, such as Resnet34, Resnet50, and Inceptionv3. These models would leverage the knowledge learned from large datasets like ImageNet. This could speed up convergence and improve generalization, further preventing overfitting.