Assignement 3

Yassine ZAOUI

yassine.zaoui@ensta-paris.fr

1. Baseline Model and Initial Results

The baseline model consisted of a simple neural network, which exhibited heavy underfitting. Both test and validation accuracies increased marginally during the early epochs and plateaued at below 10%, even after 20 epochs. This indicated that the architecture was too simplistic to learn meaningful features for the classification task.

2. Improved Architectures

2.1. Adding Convolutional Layers

To enhance feature extraction, we proposed a deeper architecture with 5-6 convolutional layers. Each layer employed primarily 3×3 kernels for computational efficiency, with occasional 5×5 kernels to expand the receptive field. Output channels were doubled after each layer, starting with 32 and ending with 512, closely matching the number of classes.

2.2. Global Average Pooling and Activation Functions

The final convolutional layer was followed by global average pooling, reducing the feature dimensionality to 512 before classification. LeakyReLU activation with a negative slope of 0.001 was used to prevent dead neurons and maintain gradient flow.

2.3. Results and Challenges

Despite these changes, the model displayed overfitting: training accuracy exceeded 96%, while validation accuracy stagnated around 50%, even after 30 epochs.

3. Mitigating Overfitting:Data Augmentation and Batch Normalization

To improve generalization, we applied data augmentation techniques, including random rotations, flips, and scaling. Batch normalization was incorporated after each convolutional layer to stabilize the learning process by normalizing feature distributions.

Improved Results

These strategies significantly improved validation accuracy, which reached 63%, while training accuracy remained around 95%. However, the overfitting issue persisted.

4. Exploration of Pretrained Models

To further enhance performance, pretrained models were explored. Initially, the `DinoV2` model was explored for its state-of-the-art capabilities in feature extraction. While it effectively avoided overfitting, fine-tuning was computationally expensive and resulted in validation accuracy stagnating at approximately 50%-60%. Consequently, `ResNet-50` was chosen as an alternative. We took its architecture, replaced its fully connected layer with a new one having 500 outputs. We introduced also a dropout layer with fairly high rate 0.5 to reduce overfitting.

4.1. Training modifications

We introduced Mixup regularization to encourage generalization over overfitting, AdamW with better weight regularization and fast convergence, Cosine Annealing Warm restars to dynamically adjust the learning rate, Mixed-Precision training using `torch.cuda.amp.GradScaler`, early stopping to detect when we fail to improve the validation accuracy and finally label smoothing to the loss function to avoid being over confident predictions.

4.2. Performance

This approach achieved a better validation accuracy of roughly 70% proof of improvement. Nevertheless, no one denies some overfitting persists and we need further experimentation to mitigate this problem.

References