Exercise 4 Report: Neural Networks and Ensemble Learning

Part 1: Multi-Layer Perceptron (MLP)

1.1 Task Description

In this section, we performed a classification task based on geographical coordinates to predict country codes from longitude and latitude features. This experiment explores the optimization and architectural choices of MLPs on tabular spatial data.

1.2 Model Architecture

The MLP model followed the required structure:

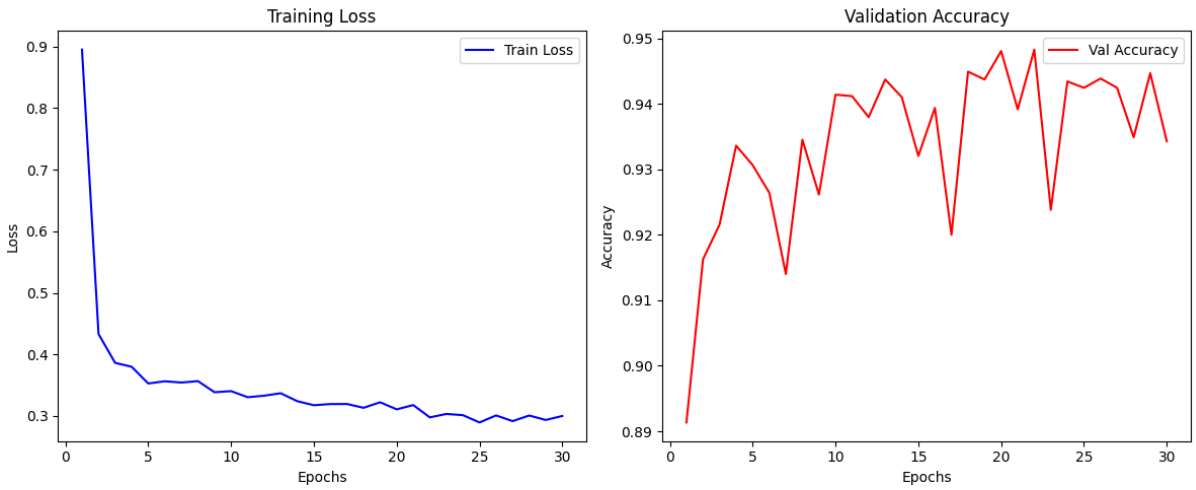
- **Input Layer:** 2 features (Longitude, Latitude).
- **Hidden Layers:** We implemented a 6-layer architecture (7 `nn.Linear` instances in total including the output layer) with 64 units each.
- **Activation:** ReLU function was used for non-linearity.
- **Regularization:** Batch Normalization was applied before each activation to stabilize training and accelerate convergence.
- **Output Layer:** Linear layer with 35 outputs corresponding to the country codes.

1.3 Optimization and Hyper-parameters (Q 6.1.2)

1. **Learning Rate Analysis:** We tested rates of 1.0, 0.01, 0.001, and 0.00001.
    - **Too High (1.0):** The model failed to converge, as large weight updates caused the loss to explode or oscillate.
    - **Too Low (0.00001):** The model learned too slowly, remaining near initial loss levels after 30 epochs.
    - **Optimal (0.001):** Provided the best balance of speed and stability.
2. **Epochs:** Training for 100 epochs showed that while more iterations allow for better boundary learning, excessive training can lead to overfitting on the specific coordinates of the training set.
3. **Batch Size:** Smaller batch sizes (e.g., 1 or 16) introduced more noise but required more iterations per epoch, while larger batches (1024) were more stable but required more epochs to reach high accuracy.

1.4 Results

- **Final Training Loss:** 0.203
- **Validation Accuracy:** 0.913
- **Test Accuracy:** 0.9133



Part 2: Convolutional Neural Networks (CNN)

2.1 Task Description

We used ResNet18 to classify real vs. AI-generated (Deepfake) faces. This task compares training from scratch, linear probing, and fine-tuning.

2.2 CNN Performance Table (Q 7.6.1)

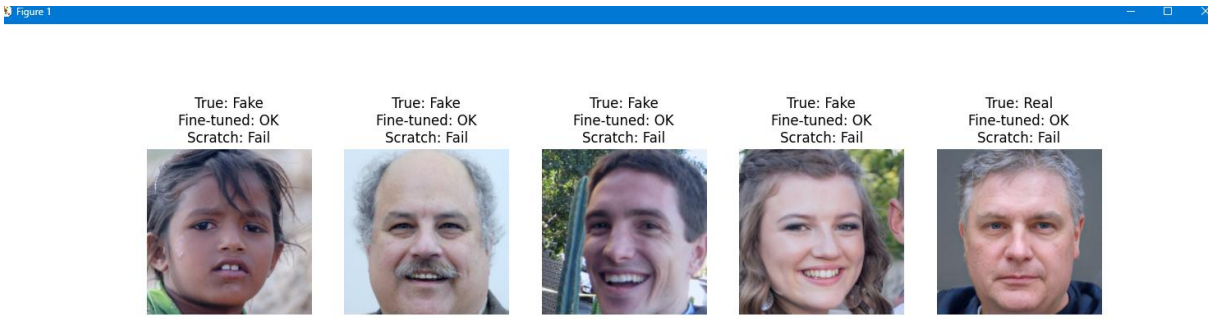
Method	Learning Rate	Test Accuracy
Training from Scratch	0.0001	0.5150
Linear Probing	0.0001	0.5150
Fine-tuning (Best)	0.0001	0.8750

2.3 Discussion of Trends

- **Fine-tuning Performance:** This baseline performed significantly better (87.5%) because it leverages pre-trained features (edges, textures) from ImageNet and adapts all layers to the specific nuances of human faces.
- **Scratch vs. Probing:** Training from scratch for only 1 epoch was insufficient for the model to learn complex facial features. Linear probing failed because the frozen ImageNet features were not specific enough for deepfake detection without further adaptation of the backbone.

2.4 Visualization (Q 7.6.2)

Below are samples correctly identified by Fine-tuning but missed by the Scratch baseline:



Part 3: XGBoost

3.1 Methodology

Images were flattened into 1D vectors and fed into an XGBoost classifier with default parameters.

3.2 Results

- **Final Test Accuracy:** 0.7375
- **Comparison:** XGBoost performed **lower** than the Fine-tuned CNN. While effective for tabular data, it lacks the spatial "Inductive Bias" of convolutional layers necessary for high-fidelity image classification.

Part 4: Summary and Conclusions

4.1 Final Comparison Table

Model	Dataset	Final Test Accuracy
MLP	Geography (CSV)	91.33%
ResNet18 (Fine-tuning)	Images	87.50%
XGBoost	Images	73.75%

4.2 Lessons Learned

- **Tabular Data:** MLPs are highly effective for low-dimensional data like coordinates.
- **Transfer Learning:** For image tasks with limited data/time, fine-tuning a pre-trained model is vastly superior to training from scratch.

- **Gradients:** In very deep networks, we must be wary of vanishing gradients; solutions like ReLU and Batch Normalization are essential for successful optimization.