Deepfake Audio Detection using RES-EfficientCNN

Objective:

The goal of this project is to detect deepfake audio from the ASVspoof 2019 Logical Access (LA) dataset using machine learning and deep learning techniques. Initially, a basic CNN architecture was explored, but due to suboptimal performance, the project progressed to using a more advanced architecture, **RES-EfficientCNN**, for better detection accuracy.

Dataset Preparation:

The dataset used in this project is the **ASVspoof 2019 Logical Access (LA) dataset**, containing both real and spoofed audio files. Our original dataset had significantly more spoof files (22,800) compared to bonafide files (2,580). This imbalance could lead to biased model training, where the model might become overly sensitive to the majority class (spoof) and underperform on the minority class (bonafide). By augmenting the bonafide files, we increased their number from 2,580 to 7,422. Using data augmentation, we improved our dataset's balance and size, which is expected to enhance the performance and reliability of our CNN model in detecting deepfake audio.

Specifically, you used a subset of **14922 audio samples**, where **Mel-spectrograms** of the audio files were generated to serve as inputs to the deep learning model.

- **Spectrogram resolution:** Original dimensions of the spectrograms are **904 x 370 pixels**.
- **Total number of files:** Around **14922** spectrogram images.
- **File format:** `.png` for the spectrogram images, `.flac` for audio files.
- **Labels:** Binary labels were assigned to audio files (Real = 0, Spoofed = 1), provided in a CSV file called `updated_labels.csv`.

Modeling Approaches

1. Initial CNN Approach:

The initial approach for this deepfake audio detection project involved using a basic **Convolutional Neural Network (CNN)** architecture to classify spectrogram images as real or spoofed audio. While the model showed some potential, it encountered several challenges during training and evaluation, particularly on the test set.

After training, the model was evaluated on the test set. The accuracy obtained on the test data was **50.26%**, which is close to random guessing for a binary classification problem (where 50% accuracy could be achieved without learning any patterns from the data). Given the performance results, it became clear that the initial CNN approach was insufficient for the task, as it barely outperformed random guessing. Thus, an alternative approach was needed.

2. Switching to RES-EfficientCNN:

In light of the suboptimal performance of the basic CNN, a more sophisticated model architecture was adopted: **RES-EfficientCNN**, which combines the strengths of residual connections (ResNet) and the EfficientNet family.

Why RES-EfficientCNN?

- **EfficientNet's strength** lies in its balanced scaling of model dimensions (depth, width, resolution), allowing for better utilization of computational resources.
- **Residual connections** from ResNet help the model avoid vanishing gradient problems, enabling it to train deeper networks more effectively.
- **Transfer Learning:** By leveraging **pre-trained EfficientNet-B0 weights**, the model could benefit from learning representations already trained on large datasets (ImageNet), which drastically improved the performance compared to training from scratch.

The spectrogram images were preprocessed before being input into the CNN model:

- **Image Transformations:**
  The spectrogram images were converted to RGB and transformed into tensors using transforms.Compose([transforms.ToTensor()]).

- **Dataset Splitting:**
  The dataset was split into:

  - **Training set:** 70%
  - **Validation set:** 15%
  - **Test set:** 15%

- **Batch Size:** A batch size of 16 was used during both training and evaluation.

Training with RES-EfficientCNN:

The RES-EfficientCNN model was trained for **10 epochs** using the following process:

- **Forward Pass:** Input images (spectrograms) were passed through the EfficientNet-based architecture.
- **Loss Calculation:** The cross-entropy loss function was used to compute the error between predictions and ground truth labels.
- **Backpropagation:** The error was backpropagated through the network, and the model weights were updated using the Adam optimizer.
- **Metrics:** After each epoch, **training loss** and **accuracy** were calculated to monitor model performance.

Validation:

After each epoch, the model was evaluated on the validation set to ensure that the model was not overfitting:

- **Validation Metrics:**
  The **validation loss** and **accuracy** were tracked to assess the generalization performance of the model on unseen data.

Results:

The performance of the RES-EfficientCNN model on the test set can be summarized as follows:

- **Improved Results with RES-EfficientCNN:** The switch to the RES-EfficientCNN architecture significantly improved the model's accuracy compared to the initial CNN approach. After training, the model achieved an accuracy of **99.85%**, which was a substantial improvement from the accuracy achieved by the basic CNN.
- **Final Results:**
    - **Training Accuracy:** Improved with each epoch, converging to a high level.
    - **Validation Accuracy:** The model maintained high accuracy during validation, indicating good generalization.

Predictions:

After training, the model was used to make predictions on new spectrogram images. The trained model successfully predicted whether an audio sample was real or spoofed.

Key Contributions:

1. **Feature Extraction with Mel-spectrograms:**
   By transforming audio files into Mel-spectrogram images, the model was able to capture both time-domain and frequency-domain features.
2. **Transfer Learning with EfficientNet:**
   Leveraging pre-trained EfficientNet-B0 weights significantly improved performance over the initial CNN model, reducing training time and boosting accuracy.
3. **Exploration of Multiple Approaches:**
   The project initially explored a basic CNN approach, but recognizing the limitations in accuracy, the advanced RES-EfficientCNN model was implemented, which yielded far better results.

Conclusion:

This project successfully implemented a **deepfake audio detection system** using the **RES-EfficientCNN** model, which achieved significantly higher accuracy than a simpler CNN model. The use of **EfficientNet-B0** enabled the model to effectively learn the complex patterns in Mel-spectrograms, ultimately achieving a high test accuracy on the ASVspoof 2019 dataset.