UNIVERSITY OF CALIFORNIA
Los Angeles

Identifying Deepfake Audio with Statistical and Deep Learning Methods

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Applied Statistics and Data Science

by

Derek Wen

2026

ABSTRACT OF THE THESIS

Identifying Deepfake Audio with Statistical
and Deep Learning Methods

by

Derek Wen
Master of Applied Statistics and Data Science
University of California, Los Angeles, 2026
Professor Ying Nian Wu, Chair

As deepfake audio becomes increasingly realistic, the need for reliable detection models is critical. This thesis evaluates the effectiveness of statistical versus deep learning methods for identifying deepfake audio in the for-norm release of the Fake-or-Real dataset. Statistical baselines are established using Logistic Regression and Random Forest models trained on handcrafted, non-learned acoustic features. The Random Forest model achieves a strong test accuracy of 0.815 and a test AUC of 0.888, showing that classical signal processing features remain competitive. Deep learning models are evaluated using log-Mel spectrogram inputs and convolutional architectures, including a lightweight CNN trained from scratch and a ResNet18 model fine-tuned for single-channel spectrograms. Threshold-based metrics are computed using a decision threshold selected on the validation set and then fixed for test evaluation on the held-out test set. Using this framework, the CNN attains a test accuracy of 0.745 with a test AUC of 0.882, while ResNet18 achieves the highest test AUC of 0.964 and a test AP of 0.963 but a lower test accuracy of 0.528 due to calibration mismatch between the validation and test splits. The results show that tree-based statistical baselines remain competitive in accuracy under fixed decision rules, while deeper architectures improve ranking performance and capture spectro-temporal cues that are not fully represented by handcrafted features, thereby highlighting the importance of calibration and threshold selection in deep learning models for reliable deployment.

The thesis of Derek Wen is approved.

Nicolas Christou

Guang Cheng

Ying Nian Wu, Committee Chair

University of California, Los Angeles

2026

To my family who acted as my
foundation and unwavering support
Thank you.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Recent advancements in generative artificial intelligence have fundamentally changed the landscape of audio generation. Technologies such as neural text-to-speech, voice conversion, and audio inpainting are now capable of producing synthetic audio that is nearly indistinguishable from genuine human recordings. While these models enable valuable applications including accessibility, virtual assistants, and automated dubbing, their increasing use also introduces significant risks related to misinformation, identity theft, and fraud. The widespread public accessibility of these models underscores an urgent need for reliable methods to detect deepfake audio.

This thesis evaluates statistical and deep learning methods for distinguishing real from fake audio using the for-norm release of the Fake-or-Real dataset, a large-scale dataset for deepfake audio detection research. The study begins with an exploratory data analysis to identify acoustic characteristics that differentiate genuine human speech from deepfake audio, examining factors such as spectral shape, energy distribution, and cepstral variability to uncover systematic patterns and potential sources of bias. The thesis then establishes interpretable statistical baselines using handcrafted acoustic features and builds deep learning models to capture complex temporal and spectral structure that manual feature engineering often misses. The main objective is to compare these approaches in terms of accuracy, ranking performance, and calibration, clarifying the relative contributions of learned and non-learned representations and identifying the conditions under which each class of model is most suitable for deployment.

CHAPTER 2

Literature Review

The methods for detecting deepfake audio have changed significantly over time, moving from classical signal processing to modern deep learning. This section reviews how these methods have evolved and provides the background for the statistical and deep learning models used in this thesis.

2.1 Classical Approaches and Feature Engineering

Early research in audio forensics relied on manual feature extraction to find artifacts created by text-to-speech systems. Researchers have long used Mel-Frequency Cepstral Coefficients as a standard tool for verifying speakers and detecting spoofing attacks. Hamza et al. [2] showed that using Mel-Frequency Cepstral Coefficients with classifiers like Support Vector Machines or Random Forests creates a strong baseline for distinguishing real audio from fake. These features capture the "timbre" or quality of the sound, which often contains small errors in older deepfakes. However, statistical methods often struggle to detect newer, more advanced attacks. This was highlighted in the ASVspoof 2019 challenge [5], which showed that baseline systems struggled to generalize to audio generated by advanced neural networks.

2.2 Deep Learning Approaches and Representation Learning

With the rise of Convolutional Neural Networks, the focus shifted from calculating features manually to letting the model learn them from data. Research by Alzantot et al. [1] showed

that treating audio as visual data or spectrograms allows Convolutional Neural Networks to learn complex patterns in time and frequency that manual features might miss. Residual Networks have become a standard choice in this field. For example, Wang and Yamagishi [6] used Residual Network architectures to benchmark various neural spoofing countermeasures, demonstrating the effectiveness of these architectures for detecting synthetic audio artifacts.

Recent surveys by Yi et al. [7] note that while deep learning models perform very well on specific datasets, they tend to overfit. They often memorize background noise instead of learning to spot the actual fake audio artifacts, showing why strong regularization techniques are necessary.

2.3 Generalization Strategies and Data Augmentation

To address the overfitting problem in deep learning, data augmentation is critical. The SpecAugment technique, introduced by Park et al. [4] for speech recognition, randomly masks blocks of time and frequency in spectrograms. This forces the model to use partial cues rather than memorizing the whole audio file. Studies have shown that such strategies can improve generalization in deepfake detection by reducing reliance on dataset-specific cues [3].

These generalization strategies, with SpecAugment in particular, motivate the regularization choices for the deep learning models evaluated in this thesis and frame their comparison with the classical feature-based statistical baselines.

CHAPTER 3

Data and Exploration

3.1 Data Collection

Data for the thesis is derived from the for-norm release of the Fake-or-Real dataset. This dataset aggregates utterances from various text-to-speech systems and human-speech corpora. The for-norm version balances labels and gender while normalizing sample rate, volume, and channel count. Each audio clip is accompanied by metadata identifying its label, source, and data split. Table 3.1 details the exact distribution of samples across the training, validation, and testing sets used in this thesis.

Table 3.1: Distribution of audio samples across data splits.

Split	**Real**	**Fake**	**Total**
Training	25,481	25,481	50,962
Validation	4,809	5,397	10,206
Test	2,264	2,370	4,634
Total	**32,848**	**32,954**	**65,802**

3.2 Data Cleaning and Filtering

To limit atypical clip lengths and reduce length-driven leakage, duration thresholds are learned exclusively on the training split using the interquartile range (IQR) rule. Let Q_1 and Q_3 be the 25th and 75th percentiles of training durations, respectively, and $\text{IQR} = Q_3 - Q_1$.

The bounds are computed as:

$$\text{lower} = \max\big(0.10,\ Q_1 - 1.5\,\text{IQR}\big), \qquad \text{upper} = Q_3 + 1.5\,\text{IQR}.$$

The lower bound is clamped at 0.10 seconds to ensure no empty or near-empty files are processed. Utterances with a duration in the interval $(\text{lower}, \text{upper}]$ are retained across training, validation, and testing splits. Audio clips falling outside these bounds are omitted to ensure model stability. Lastly, a check confirmed that no null values or duplicate entries existed in the dataset.

3.3 Feature Extraction

Feature extraction is performed on the cleaned dataset using the librosa library, with all audio processed at a consistent sampling rate of 16 kHz. To establish a statistical baseline, the raw audio waveforms are transformed into fixed-length feature vectors through a sequence of signal processing steps. This process extracts both physical descriptors, which capture the energy and shape of the spectrum, and perceptual cepstral coefficients, which characterize timbre.

3.3.1 Framing and Windowing

To analyze non-stationary audio signals, the continuous waveform is sliced into short, overlapping frames. For the statistical baseline features, a window size of 2048 samples (approximately 128ms) was used to capture broader spectral texture, whereas the deep learning models utilized finer 25ms windows. To minimize spectral leakage at the edges of these frames, a Hann window $w(n)$ is applied to each frame:

$$w(n) = 0.5\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right) \tag{3.1}$$

where N is the number of samples in the window.

3.3.2 Spectral Transformation

Following windowing, the transition from the time domain to the frequency domain is performed using the Discrete Fourier Transform (DFT). For a windowed signal $w(n)$ of length N, the DFT is defined as:

$$X(k) = \sum_{n=0}^{N-1} w(n) e^{-j\frac{2\pi}{N}kn}, \quad 0 \leq k \leq N-1 \tag{3.2}$$

where $X(k)$ represents the complex spectral magnitude at frequency bin k.

To align the feature representation with human auditory perception, the power spectrum derived from the DFT is mapped onto the Mel scale. The mapping from frequency f (Hertz) to Mel m is typically approximated by:

$$m = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right) \tag{3.3}$$

This logarithmic transformation allocates higher resolution to lower frequencies (0–1000Hz), where human speech characteristics are most dense, while compressing higher frequencies.

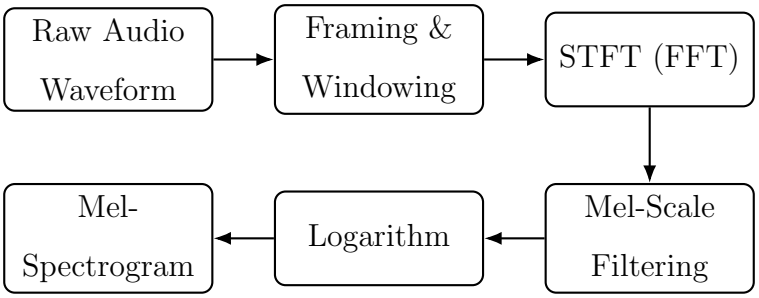


Figure 3.1: Raw audio is converted into a time–frequency representation through a sequence of signal processing steps, mimicking human auditory perception.

3.3.3 Derived Acoustic Features

From this transformed spectral data, two categories of features are derived to form the final dataset for the statistical models.

3.3.3.1 Mel-Frequency Cepstral Coefficients

To capture the "timbre" or identity of the sound source, 13 Mel-Frequency Cepstral Coefficients (MFCCs) are extracted. This is achieved by applying the Discrete Cosine Transform (DCT) to the log-Mel spectrum $S(m)$:

$$c_n = \sum_{m=0}^{M-1} S(m) \cos\left(\frac{\pi n(m+0.5)}{M}\right) \tag{3.4}$$

where M is the number of Mel filterbanks. These coefficients are summarized by both their means and standard deviations across the time axis to capture timbral consistency and variability, resulting in 26 cepstral features per clip.

3.3.3.2 Spectral and Physical Descriptors

In addition to cepstral features, specific spectral descriptors are calculated to capture physical characteristics. Let $X(k)$ be the magnitude of the spectrum at frequency bin k, and $f(k)$ be the center frequency of that bin.

- **Spectral Centroid:** Represents the "brightness" or center of mass of the spectrum.

$$\mu_{\text{spec}} = \frac{\sum_k f(k) \cdot |X(k)|}{\sum_k |X(k)|} \tag{3.5}$$

- **Spectral Bandwidth:** Measures the width of the frequency band.

$$\sigma_{\text{spec}} = \sqrt{\frac{\sum_k (f(k) - \mu_{\text{spec}})^2 \cdot |X(k)|}{\sum_k |X(k)|}} \tag{3.6}$$

- **Spectral Rolloff:** The frequency below which a specified percentage κ of the total spectral energy lies. It is defined as the frequency bin K satisfying:

$$\sum_{k=0}^{K} |X(k)| \geq \kappa \sum_{k=0}^{N/2} |X(k)| \tag{3.7}$$

- **Root-Mean-Square (RMS) Energy:** Represents the loudness or amplitude of the signal. For a signal $x(n)$ of length N:

$$\text{RMS} = \sqrt{\frac{1}{N} \sum_{n=0}^{N-1} |x(n)|^2} \tag{3.8}$$

- **Zero-Crossing Rate (ZCR):** The rate at which the signal changes sign.

$$\text{ZCR} = \frac{1}{N-1} \sum_{n=1}^{N-1} \mathbb{I}(x(n)x(n-1) < 0) \tag{3.9}$$

  where $\mathbb{I}$ is the indicator function.

These time-series descriptors are summarized by their means over the duration of the clip. The combination of aggregated MFCCs and spectral descriptors results in a fixed-length feature vector suitable for baseline statistical classification.

3.4 Exploratory Data Analysis

The exploratory data analysis begins by examining the distribution of audio duration, a potential confounding variable. Figure 3.2 illustrates the duration distribution after cleaning. Real utterances span a much wider range of lengths, whereas fake utterances concentrate around shorter segments of roughly 1.2–2.0 seconds. This separation indicates that duration alone could yield strong apparent performance.

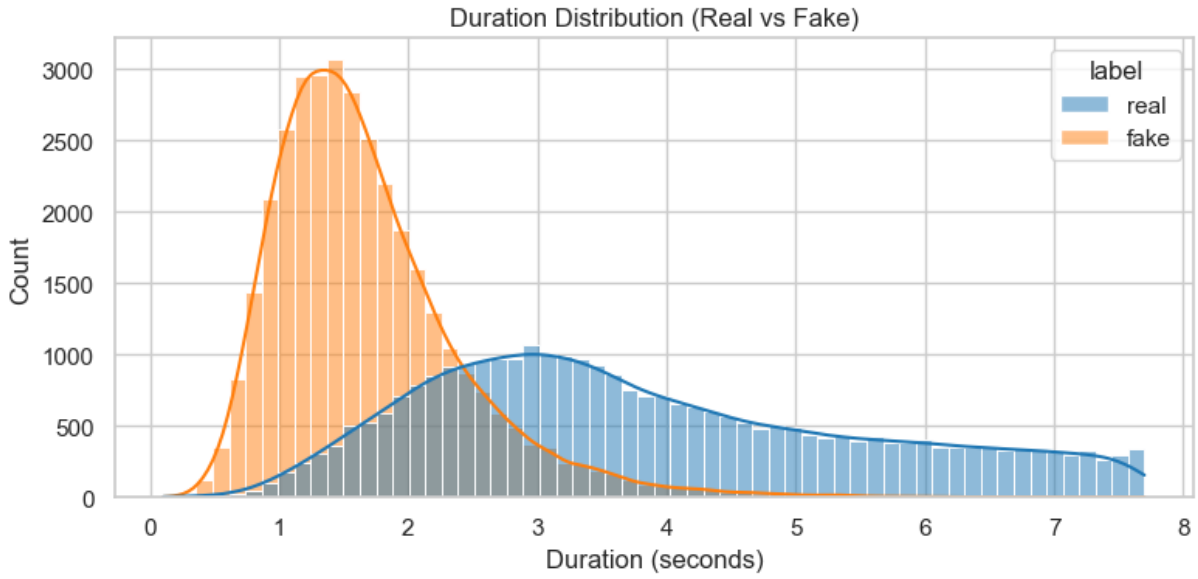


Figure 3.2: Duration distribution by label after cleaning.

Specific acoustic descriptors are further analyzed in Figure 3.3. RMS energy is shifted

higher for fake audio despite dataset-level normalization, suggesting that normalization does not fully eliminate energy differences introduced by different synthesis pipelines. Spectral centroid and bandwidth are consistently higher for real audio, consistent with the naturally richer high-frequency content found in human recordings compared to band-limited synthesis. Zero-crossing rate differs only slightly, suggesting it is unlikely to be a primary predictive factor.

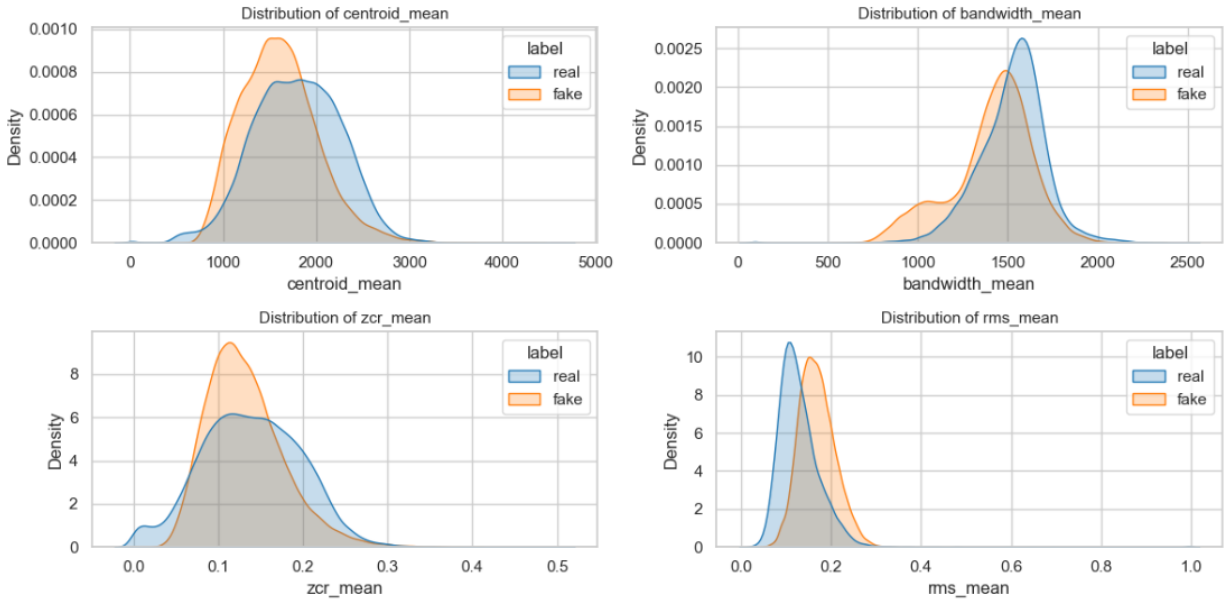


Figure 3.3: Kernel Density Estimates for centroid, bandwidth, ZCR, and RMS by label.

The boxplots in Figure 3.4 confirm that these spectral shifts are not driven by outliers. Medians and interquartile ranges are consistently higher for real audio on centroid, bandwidth, and rolloff, while fake audio tends to be louder, exhibiting higher RMS. However, significant overlap remains, indicating that no single feature is sufficient for classification.

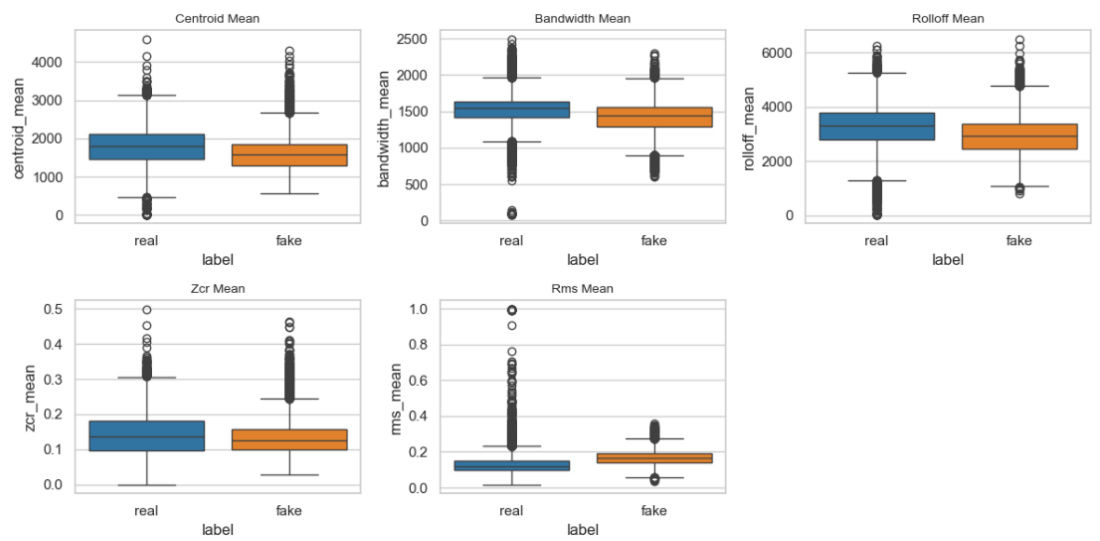


Figure 3.4: Boxplots for selected features by label.

To validate these statistical observations, the spectral centroid overlay is visualized on sample spectrograms from both classes. As shown in Figure 3.5, the real audio exhibits a centroid that reaches higher overall frequencies, indicated by the white line, in contrast with the synthetic sample that displays a lower-frequency centroid trajectory. This visual confirmation aligns with the boxplot distributions, suggesting that the generative models in this dataset fail to perfectly reconstruct the high-frequency complexity of natural speech.

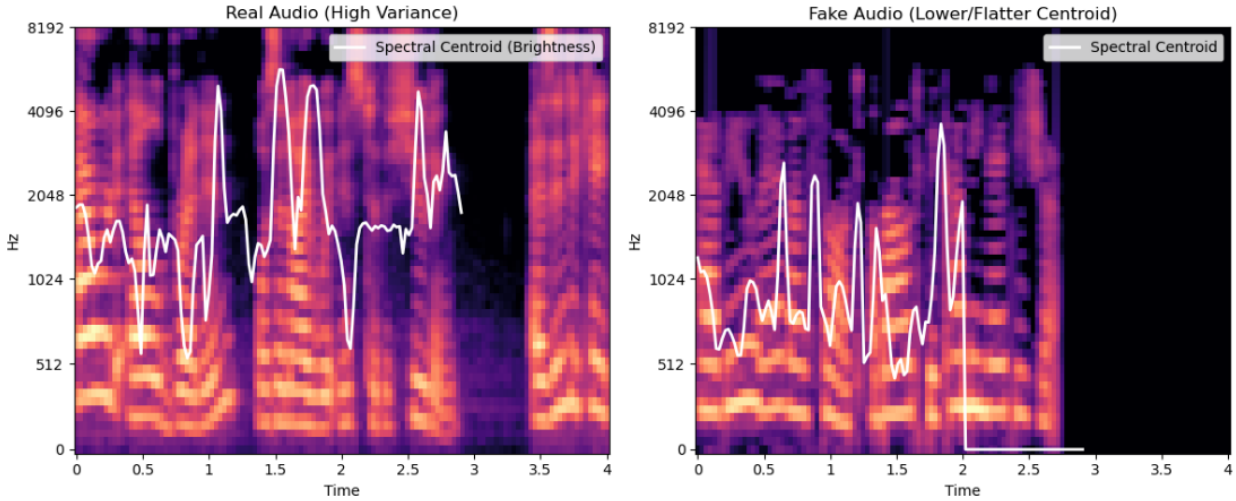


Figure 3.5: Spectral Centroid Overlay. The white line tracks the center of mass of the spectrum over time.

To assess feature redundancy, the correlation matrix in Figure 3.6 shows that duration has the largest positive association with the real label, while RMS has the largest negative association. Spectral shape features such as centroid, bandwidth, and rolloff and several MFCC statistics also correlate with the label but with smaller magnitudes. This suggests they still provide useful signal beyond duration and loudness, even though these two variables dominate the overall separation.

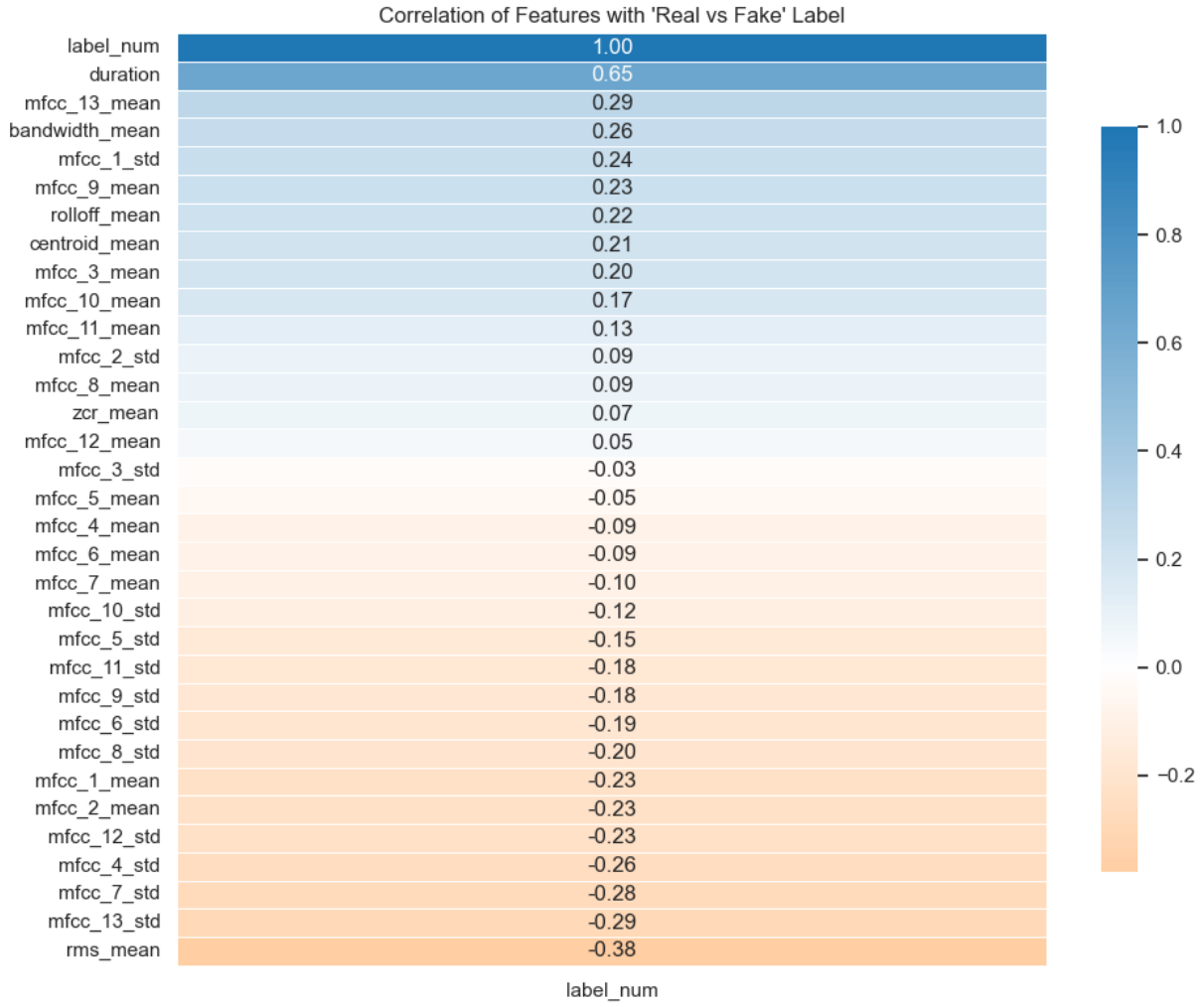


Figure 3.6: Correlation of numeric features with label_num (real=1, fake=0).

3.5 Feature Selection

Taken together, the exploratory data analysis shows that the most stable distinctions emerge from spectral and cepstral characteristics, including higher spectral centroids, broader bandwidths, and consistent MFCC patterns that reflect natural speech variability. In contrast, the zero-crossing rate exhibits limited discriminative ability, indicating that higher-level spectral and temporal features are more informative than simple time-domain measures.

Duration and loudness dominate the apparent separation between real and fake clips, but their unusually strong influence suggests that models could exploit them as shortcuts rather than learning more generalizable acoustic cues. Duration is largely an artifact of data processing, whereas loudness reflects a genuine property of the synthesis process. Therefore, duration was excluded from all subsequent modeling steps, while loudness was retained.

With these considerations in place, the remaining features were processed to improve model stability and interpretability. To reduce multicollinearity, pairwise Pearson correlations were computed, and features exhibiting a coefficient greater than 0.95 were automatically pruned. Finally, prior to model training, all extracted feature vectors were standardized to zero mean and unit variance using scaling parameters fit on the training split and then applied to the validation and testing splits to ensure that each feature contributed on a comparable scale.

CHAPTER 4

Statistical Baseline

This chapter establishes baseline statistical models to evaluate their performance and the information captured by handcrafted acoustic features before introducing deep learning methods.

4.1 Methodology

The statistical baselines use the handcrafted features described in the previous chapter, representing each audio clip as a fixed-length numerical vector. Logistic Regression is trained with an ℓ_2 penalty, a maximum of 1000 iterations, and class-balanced weights to avoid sensitivity to mild imbalance across splits. Random Forest is trained with 400 trees, a minimum leaf size of 2, and class-balanced weights. Both models are fit on the training split only. For evaluation, each model outputs a probability that a clip is real on the validation and test splits. Class labels used to compute accuracy, precision, and recall are obtained using the model's default decision rule. Threshold-free metrics include the area under the ROC curve (AUC) and average precision (AP), which are computed directly from the predicted probabilities.

Figure 4.1 summarizes the statistical baseline pipeline. Handcrafted features are standardized and pruned, passed to Logistic Regression and Random Forest to produce real-class probabilities, and then converted to class labels and evaluation metrics.

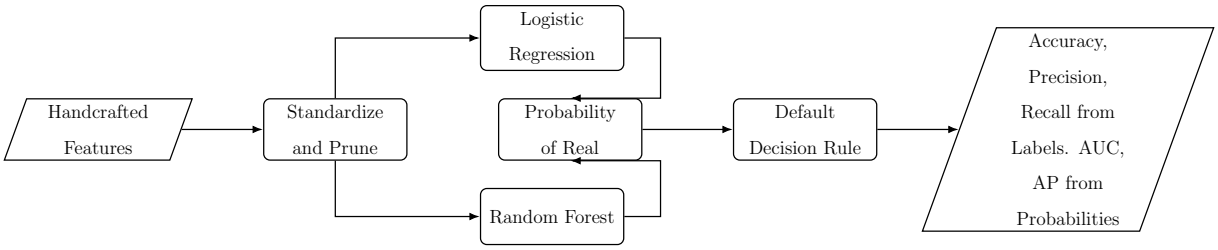


Figure 4.1: Statistical baseline pipeline for deepfake audio detection.

4.1.1 Logistic Regression

A Logistic Regression model provides an interpretable linear benchmark. The model estimates the probability that an input feature vector $\mathbf{x}$ belongs to the "Real" class ($y = 1$) using the sigmoid function:

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + e^{-(\beta_0 + \boldsymbol{\beta}^T \mathbf{x})}} \tag{4.1}$$

where $\boldsymbol{\beta}$ represents the vector of learned coefficients corresponding to the acoustic features, and β_0 is the intercept term. The optimal parameters $\boldsymbol{\beta}$ are learned by minimizing a regularized negative log-likelihood (Log-Loss) over the training set of N samples:

$$J(\boldsymbol{\beta}) = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \; + \; \frac{\lambda}{2} \|\boldsymbol{\beta}\|_2^2, \tag{4.2}$$

where $\lambda > 0$ controls the strength of the ℓ_2 regularization.

4.1.2 Random Forest

While Logistic Regression provides a linear baseline, audio data often contains complex, nonlinear interactions. To capture these, a Random Forest classifier is employed. The quality of a split in each decision tree node t is determined by the Gini Impurity $I_G(t)$:

$$I_G(t) = 1 - \sum_{i=1}^{C} p_i^2 \tag{4.3}$$

where p_i is the probability of an item belonging to class i and C is the total number of classes (here, $C = 2$). The final prediction is obtained by majority voting across all trees in the

ensemble. The split criterion maximizes the Information Gain (ΔI), which represents the reduction in impurity achieved by splitting a node N into child nodes N_L and N_R:

$$\Delta I = I_G(N) - \left(\frac{|N_L|}{|N|} I_G(N_L) + \frac{|N_R|}{|N|} I_G(N_R) \right) \tag{4.4}$$

This ensures that the resulting child nodes are as homogeneous (pure) as possible.

4.2 Results

4.2.1 Performance

This section reports the statistical baseline performance. The results are summarized in Table 4.1.

Table 4.1: Validation and test performance of baseline models.

Model	Split	Accuracy	AUC	AP	Precision	Recall
Logistic Regression	Validation	0.87	0.93	0.91	0.85	0.87
Random Forest	Validation	0.98	0.99	0.99	0.98	0.96
Logistic Regression	Test	0.61	0.65	0.65	0.64	0.47
Random Forest	Test	0.82	0.89	0.90	0.84	0.76

As shown in Table 4.1, both Logistic Regression and Random Forest demonstrate that the extracted acoustic features provide a strong indicator for distinguishing real from fake audio. Logistic Regression achieves strong validation metrics but suffers a significant drop in test AUC, indicating that the linear relationships it learned do not generalize well to unseen data. Random Forest achieves near-perfect validation scores and retains strong test results, indicating that non-linear feature dependencies improve generalization to unseen data.

Figure 4.2 shows the receiver operating characteristic (ROC) curves on the held-out test set, revealing that Random Forest achieves substantially higher discrimination ability compared with Logistic Regression. The Random Forest ROC curve remains consistently

above the Logistic Regression curve across most false positive rates, indicating that non-linear interactions provide more reliable separation between real and fake clips across operating points than the linear baseline.

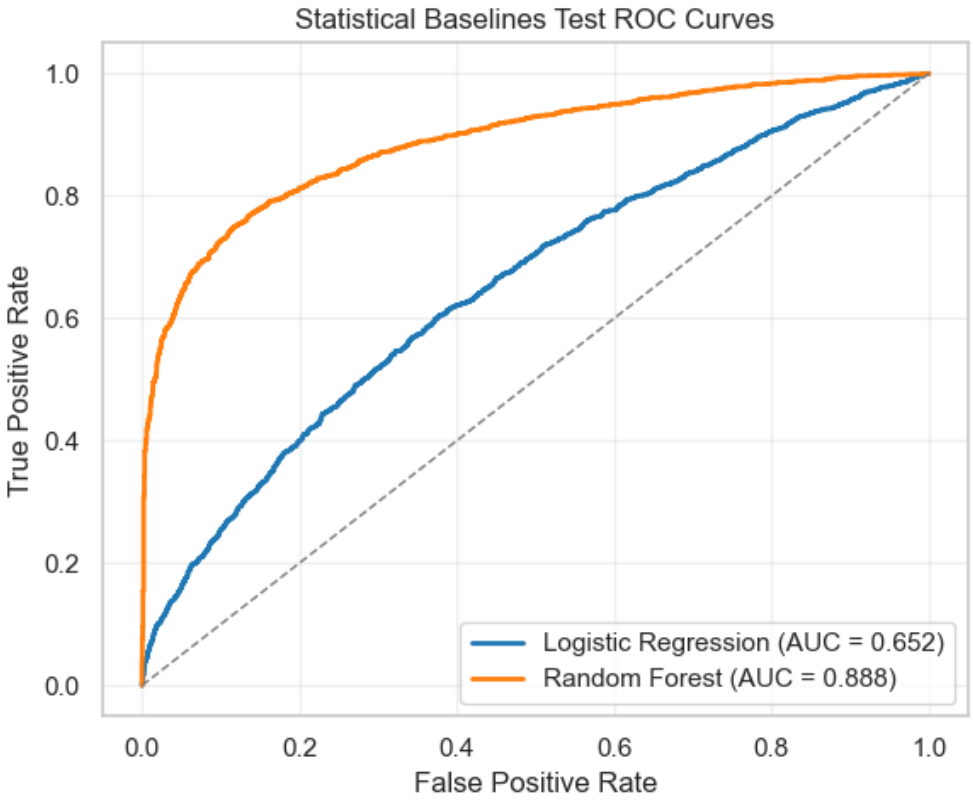


Figure 4.2: ROC curves for Logistic Regression and Random Forest on the test set.

Figure 4.3 presents the precision–recall curve on the test set and shows that Random Forest again outperforms Logistic Regression, with higher average precision. Random Forest maintains higher precision across a wide range of recall values, while Logistic Regression shows a sharper drop in precision as recall increases. This suggests that the non-linear model ranks real clips ahead of fake clips more effectively when high recall is needed.

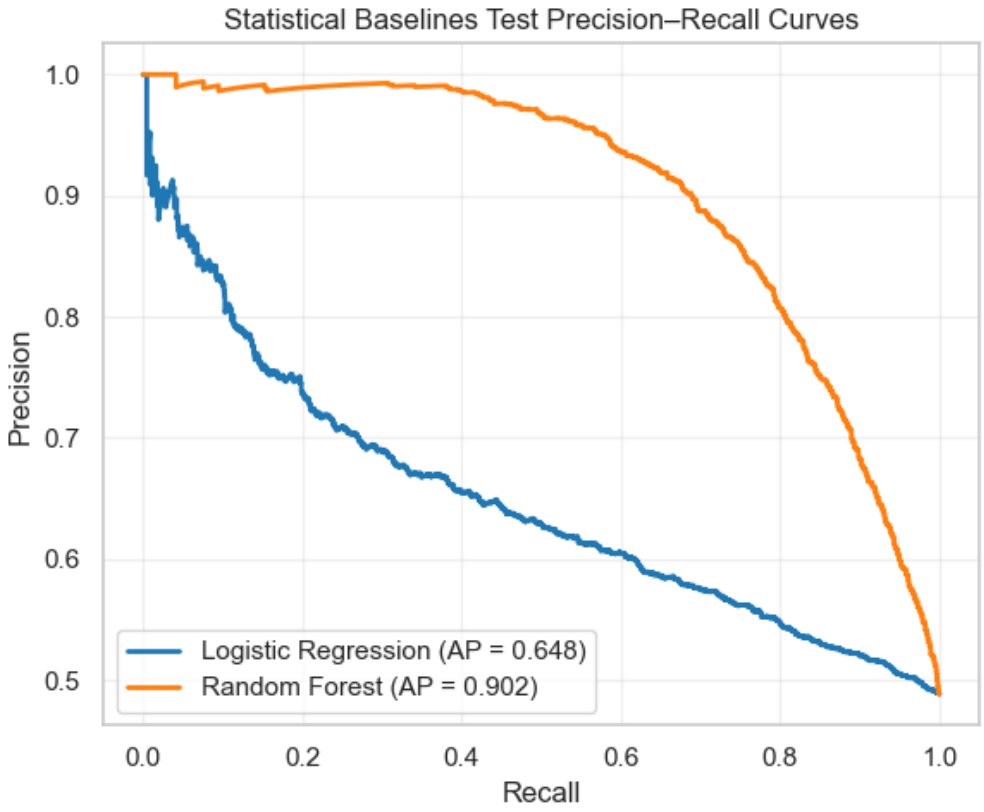


Figure 4.3: Precision–Recall curves for Logistic Regression and Random Forest on the test set.

Figure 4.4 presents the calibration curves on the test set and shows that while Random Forest is more accurate, it is less well calibrated than Logistic Regression and often outputs extreme probabilities even when incorrect. This means that predicted probabilities from the Random Forest do not match true likelihoods as closely as those from Logistic Regression. For this reason, its probability scores should be interpreted with caution, especially when choosing a decision threshold.

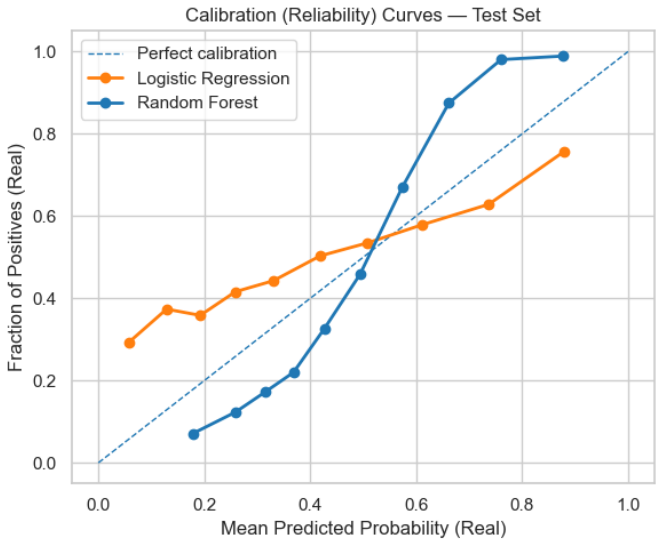
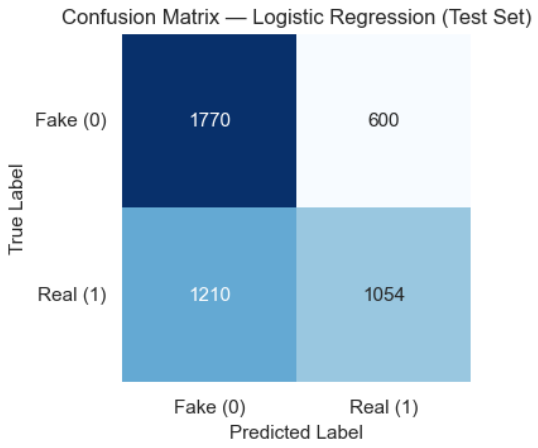
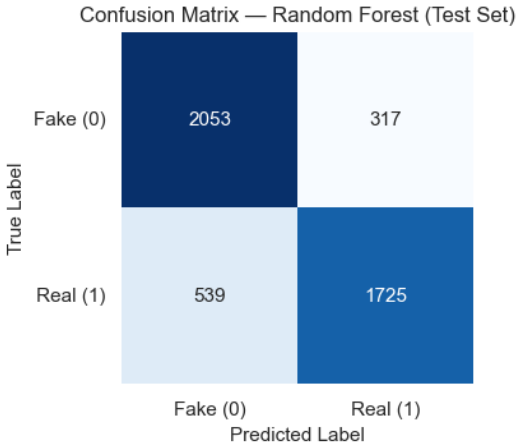


Figure 4.4: Calibration curves for baseline models. Random Forest shows significant overconfidence (S-shape), pushing probabilities to extremes.

The discrepancy between validation and test performance is further analyzed via confusion matrices on the test set in Figure 4.5. Logistic Regression exhibits a high false negative rate, frequently misclassifying real clips as fake, while Random Forest performs more evenly across both classes.



(a) Confusion Matrix of Logistic Regression on the test set.



(b) Confusion Matrix of Random Forest on the test set

Figure 4.5: Statistical Baseline Confusion Matrices on the test set.

Feature importance analysis reveals the acoustic drivers of these predictions, as Figures 4.6 and 4.7 display the top predictors. Both models highlight RMS energy, spectral centroid, and several MFCC statistics as the primary contributors to their predictions.

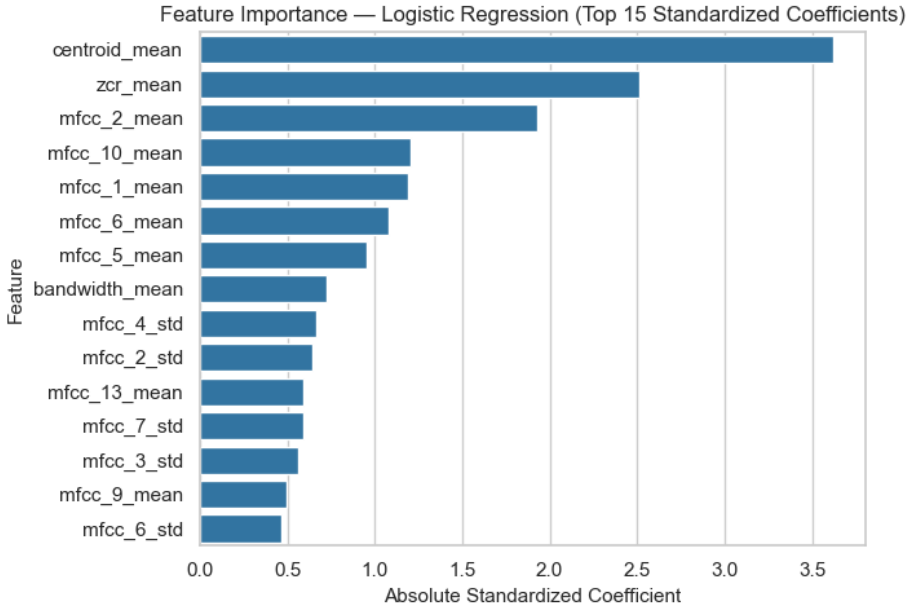


Figure 4.6: Logistic Regression coefficients.

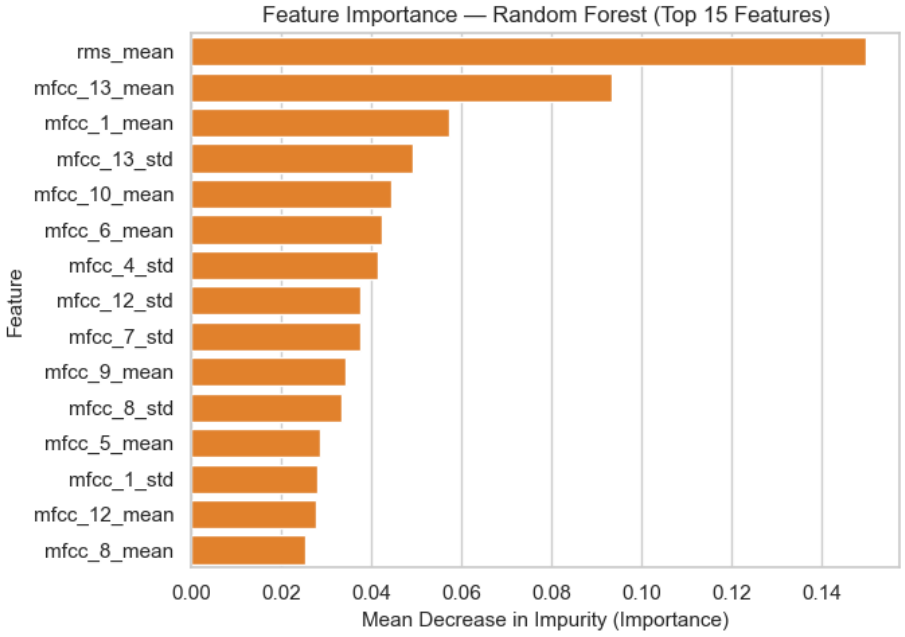


Figure 4.7: Random Forest feature importance.

4.2.2 Discussion

The baseline experiments confirm that traditional statistical models can effectively distinguish between real and fake audio using handcrafted acoustic features. However, they also reveal critical limitations in generalization.

Logistic Regression provides a transparent baseline, revealing that frequency and energy features alone carry significant predictive value. However, its test performance of 61% accuracy suggests that the boundary between real and deepfake audio in the test set is not linearly separable.

Random Forest achieves near-perfect validation accuracy and maintains high test performance. The drop in accuracy from 98% on the validation set to 82% on the test set highlights a generalization gap that may reflect distributional differences between the validation and test sets. This indicates that while statistical features capture meaningful information, they may not generalize effectively to unseen speakers, recording conditions, or generation algorithms.

These results establish a clear motivation for the deep learning stage. The baselines demonstrate that while acoustic features such as RMS, spectral centroid, and MFCCs provide stable separation, statistical summaries alone are insufficient for generalized deepfake detection.

CHAPTER 5

Deep Learning

Following the analysis of the statistical baseline models, this chapter establishes deep learning methods to evaluate their performance and to assess whether learned representations could exceed statistical baseline performance.

5.1 Methodology

The statistical baseline models relied upon global averages and collapsed the time dimension into scalar values. In contrast, the deep learning approach uses Convolutional Neural Networks (CNN) that take spectrograms as input and capture time–frequency patterns. The CNN and ResNet18 models were trained with the Adam optimizer and cross-entropy loss for 8 epochs with a batch size of 64, with SpecAugment applied during training as a form of regularization. The CNN used an initial learning rate of 10^{-3}, while ResNet18 used a lower learning rate of 10^{-4} with a weight decay of 10^{-4} for fine-tuning. All training used the training split, and the checkpoint for each model was selected using validation AUC. For threshold-based metrics, a decision threshold τ^* was selected on the validation set and then fixed for test evaluation. AUC and AP were computed from predicted probabilities and do not depend on a threshold.

Figure 5.1 summarizes the deep learning pipeline from raw audio to model outputs. Each clip is converted into a log-Mel spectrogram, normalized with a per-sample z-score transform, augmented with SpecAugment during training only, and then passed through a convolutional model (CNN or ResNet18) and binary classification head to produce a real or fake prediction.

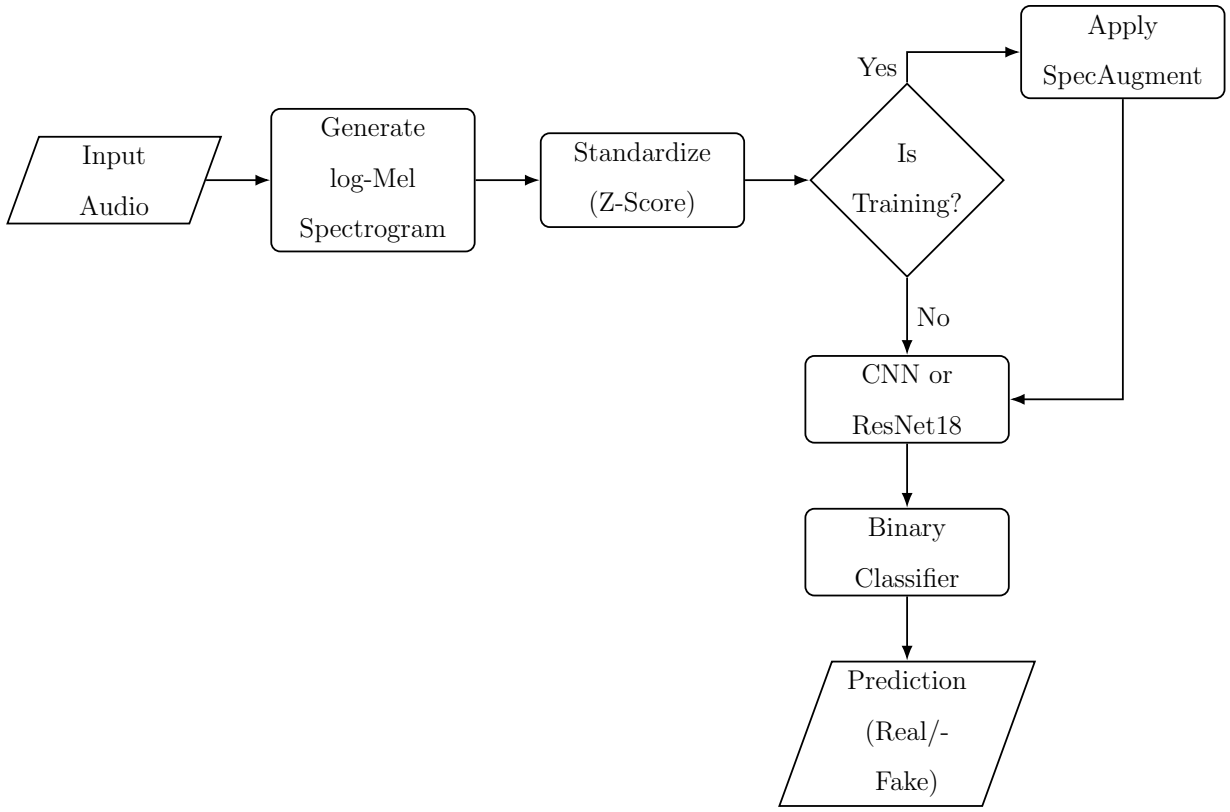


Figure 5.1: Deep learning pipeline for deepfake audio detection.

5.1.1 Model Selection Rationale

Convolutional models, specifically a CNN and ResNet18, were chosen over sequence models such as recurrent networks or attention-based Transformers for three reasons.

First, the structure of the input fits convolution well. Log-Mel spectrograms are two-dimensional time–frequency images, and important patterns often appear in small local regions. Convolutional networks are built to detect these local patterns using shared filters. In contrast, recurrent models such as LSTMs or GRUs mainly process information along the time axis, and they do not capture joint time–frequency patterns as directly without extra design.

Second, dataset size and training reliability favored convolutional models. Transformer-based approaches such as Wav2Vec 2.0 and vision-style Transformers can work well, but they

often need larger datasets or strong pretraining to avoid overfitting when trained end-to-end. The for-norm release contains roughly 66,000 samples, and the dataset used here is slightly smaller after duration filtering. In this setting, convolutional models offer a good balance of model size and structure for the task.

Third, ResNet-style skip connections help when using deeper networks. The cues that separate real and deepfake audio can be subtle, and deeper models can be harder to train. ResNet18 uses skip connections to stabilize optimization in deeper architectures and supports a higher-capacity model than the baseline CNN. For these reasons, this thesis evaluates both a lightweight CNN trained from scratch and a ResNet18 model fine-tuned for binary classification.

5.1.2 Data Representation and Pre-processing

Raw audio waveforms were converted into log-Mel spectrograms using a sampling rate of 16 kHz and a fixed duration of 4.0 seconds. The transformation from linear frequency to the perceptual Mel scale follows the same mapping introduced earlier in Equation 3.3. This representation preserves time–frequency structure while giving more detail in the lower frequency range where vocal characteristics are strongest. The spectrograms were computed using a Short-Time Fourier Transform with a window size of 400 samples, a hop length of 160 samples, and 64 Mel-Frequency bands covering 20–8000 Hz.

Clips shorter than 4.0 seconds were padded with silence. For the baseline spectrogram dataset, clips longer than 4.0 seconds were truncated to the first 4.0 seconds. For the augmented training pipeline, clips longer than 4.0 seconds were randomly cropped during training to increase diversity, while validation and test clips used a deterministic center crop.

Figure 5.2 shows an example real and fake spectrogram after this preprocessing. The black region indicates zero-padding introduced to satisfy the fixed input duration. In this example, the fake clip was shorter than 4.0 seconds and required padding, whereas the real clip was longer and did not.

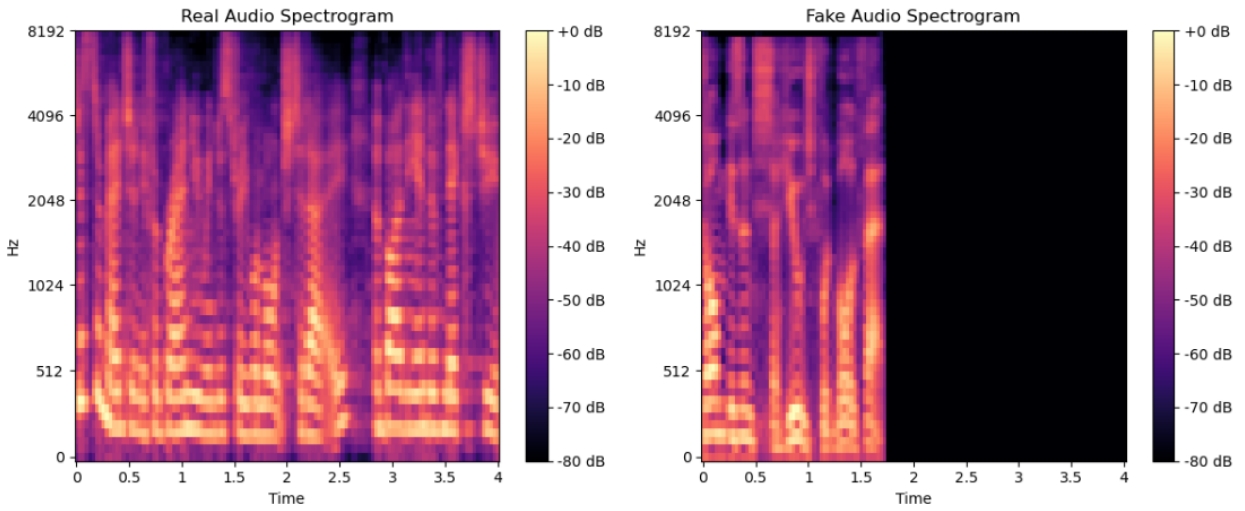


Figure 5.2: Example log-Mel spectrograms for real and fake audio after padding and cropping to 4.0 seconds.

The resulting input dimension for the models was $1 \times 64 \times 400$, which corresponds to one channel, 64 Mel-Frequency bands, and roughly 400 time frames. Each spectrogram was normalized per sample using a z-score transform to reduce sensitivity to loudness differences, addressing the loudness bias observed in the exploratory analysis. After normalization, SpecAugment was applied during training only, as described in the methodology section.

5.1.3 Convolutional Neural Network

The first deep learning model implemented was a lightweight three-layer Convolutional Neural Network. This model serves as a learning baseline to evaluate standard convolutional feature extraction on log-Mel spectrograms without residual connections.

The core operation of the network is the 2D convolution, which extracts local time–frequency patterns from the input spectrogram X. For an input at position (i, j) and a learnable kernel K, the feature map value S is computed as:

$$S(i,j) = (X * K)(i,j) = \sum_m \sum_n X(i+m, j+n) \cdot K(m,n) \tag{5.1}$$

where the summation is over the dimensions of the kernel.

Non-linearity is introduced via the Rectified Linear Unit activation function, defined as:

$$\sigma(x) = \max(0, x) \tag{5.2}$$

This function improves optimization by preventing negative activations from propagating and supports stable gradient flow in deep networks. The CNN applies stacked convolutional blocks with pooling to progressively compress the time and frequency axes, followed by a final classification layer that outputs logits for real and fake. The architectural flow is shown in Figure 5.3.

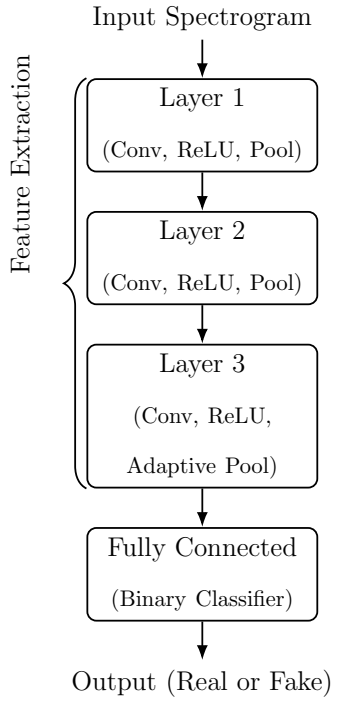


Figure 5.3: Architecture of the CNN.

To complement the high-level architectural view in Figure 5.3, Table 5.1 provides the precise layer-wise configuration. The model is designed to be lightweight, containing 23,650 trainable parameters. Max Pooling in the early layers reduces the feature map size, while the final Adaptive Average Pooling layer produces a fixed-length output vector for any input duration. This collapses the feature maps into a 64-dimensional vector.

Table 5.1: Detailed architecture of the custom 3-layer CNN. The input is a log-Mel spectrogram of size $(1, 64, 400)$.

Layer Name	**Configuration**	**Input Shape**	**Output Shape**	**Param #**
Input	—	—	$(1, 64, 400)$	0
Block 1	Conv2d (3×3, $s = 1$, $p = 1$)	$(1, 64, 400)$	$(16, 64, 400)$	160
	BatchNorm2d	$(16, 64, 400)$	$(16, 64, 400)$	32
	ReLU	—	—	0
	MaxPool2d (2×2, $s = 2$)	$(16, 64, 400)$	$(16, 32, 200)$	0
Block 2	Conv2d (3×3, $s = 1$, $p = 1$)	$(16, 32, 200)$	$(32, 32, 200)$	4,640
	BatchNorm2d	$(32, 32, 200)$	$(32, 32, 200)$	64
	ReLU	—	—	0
	MaxPool2d (2×2, $s = 2$)	$(32, 32, 200)$	$(32, 16, 100)$	0
Block 3	Conv2d (3×3, $s = 1$, $p = 1$)	$(32, 16, 100)$	$(64, 16, 100)$	18,496
	BatchNorm2d	$(64, 16, 100)$	$(64, 16, 100)$	128
	ReLU	—	—	0
	AdaptiveAvgPool (Global)	$(64, 16, 100)$	$(64, 1, 1)$	0
Classifier	Flatten	$(64, 1, 1)$	(64)	0
	Linear ($64 \to 2$)	(64)	(2)	130
Total				**23,650**

5.1.4 ResNet18

The second deep learning model employed was a ResNet18, an 18-layer residual network. The core idea of the ResNet architecture is the residual building block shown in Figure 5.4. If $\mathcal{H}(x)$ denotes the target mapping, the block learns a residual function $\mathcal{F}(x) := \mathcal{H}(x) - x$. The output y of a residual block is given by

$$y = \sigma\left(\mathcal{F}(x, \{W_i\}) + x\right) \tag{5.3}$$

where x is the input, W_i denotes the weights, and σ is the ReLU activation function.

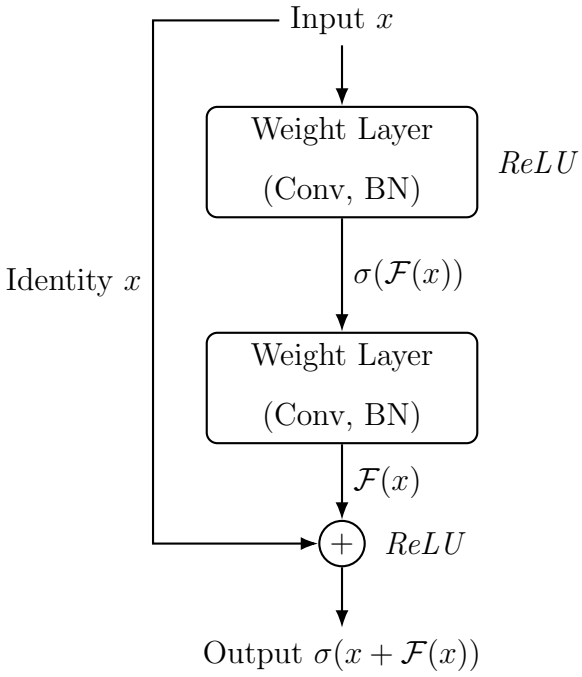


Figure 5.4: Residual block with identity skip connection x and residual function $\mathcal{F}(x)$.

To adapt ResNet18 for audio classification, the first convolutional layer was modified to accept a single input channel corresponding to the log-Mel spectrogram, and the final fully connected layer was replaced with a binary output head for real–fake classification. The model was initialized from pretrained ResNet18 weights and fine-tuned on the spectrogram task. The full network follows a four-stage residual architecture and progressively compresses the time axis from 400 frames to 13 frames before global pooling. Table 5.2 summarizes the modified architecture used in this study.

Table 5.2: Detailed architecture of the modified ResNet18 adapted for single-channel audio.

Stage	**Layer Configuration**	**Output Shape**
Input	Log-Mel Spectrogram	$(1, 64, 400)$
Stem	Conv2d (1×64, $k = 7$, $s = 2$, $p = 3$)	$(64, 32, 200)$
	BatchNorm + ReLU	$(64, 32, 200)$
	MaxPool ($k = 3$, $s = 2$, $p = 1$)	$(64, 16, 100)$
Layer 1	$\begin{bmatrix} \text{Conv } 3 \times 3, 64 \\ \text{Conv } 3 \times 3, 64 \end{bmatrix} \times 2$ blocks	$(64, 16, 100)$
Layer 2	$\begin{bmatrix} \text{Conv } 3 \times 3, 128 \\ \text{Conv } 3 \times 3, 128 \end{bmatrix} \times 2$ blocks	$(128, 8, 50)$
	Downsample (stride 2)	
Layer 3	$\begin{bmatrix} \text{Conv } 3 \times 3, 256 \\ \text{Conv } 3 \times 3, 256 \end{bmatrix} \times 2$ blocks	$(256, 4, 25)$
	Downsample (stride 2)	
Layer 4	$\begin{bmatrix} \text{Conv } 3 \times 3, 512 \\ \text{Conv } 3 \times 3, 512 \end{bmatrix} \times 2$ blocks	$(512, 2, 13)$
	Downsample (stride 2)	
Head	AdaptiveAvgPool2d (Global)	$(512, 1, 1)$
	Flatten	(512)
	Linear ($512 \to 2$)	(2)

5.1.5 Data Augmentation with SpecAugment

To reduce overfitting and encourage the model to learn cues that are spread across time and frequency, SpecAugment was used during training. SpecAugment applies time masking and frequency masking by masking contiguous regions of the log-Mel spectrogram, which limits the model's ability to rely on a small set of highly local patterns. In this setup, SpecAugment used up to 10 masked Mel bands and 30 masked time steps, with two masks applied in each direction per spectrogram. This augmentation is used for both the CNN and ResNet18 training pipelines.

Figure 5.5 illustrates the effect of this augmentation. The original spectrogram preserves all time–frequency detail, while the augmented version contains masked regions in both the time and frequency directions.

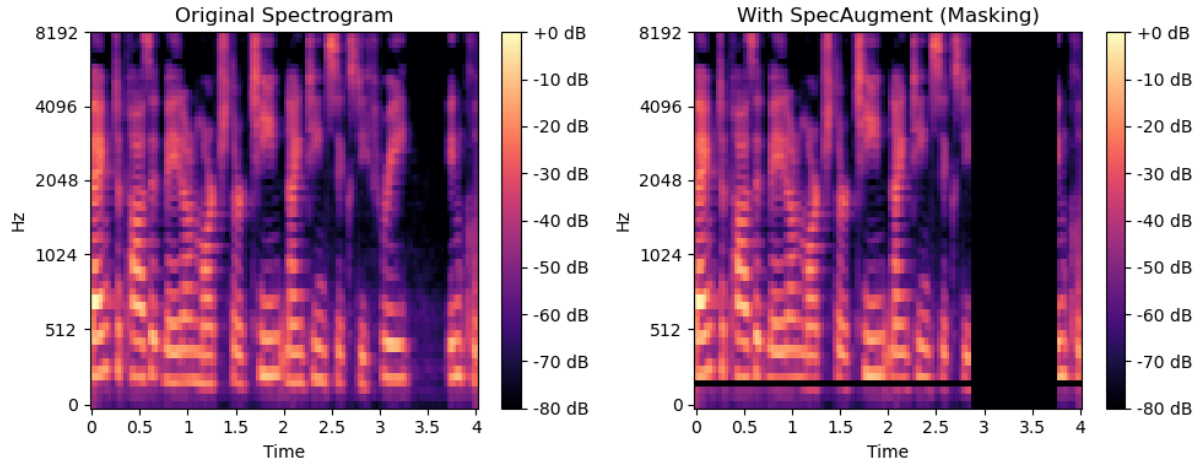


Figure 5.5: Visual demonstration of SpecAugment applied to a log-Mel spectrogram.

SpecAugment was applied only during training and was omitted during validation and testing to ensure deterministic evaluation. This encourages the model to remain effective when parts of the signal are less informative, which can occur due to noise, recording differences, or variation in speaking style.

5.2 Results

5.2.1 Performance

This section reports deep learning performance under the validation-selected decision threshold τ^*. For the CNN, the validation-selected threshold was $\tau^* = 0.400$, while for ResNet18 it was $\tau^* = 0.425$. The results are summarized in Table 5.3.

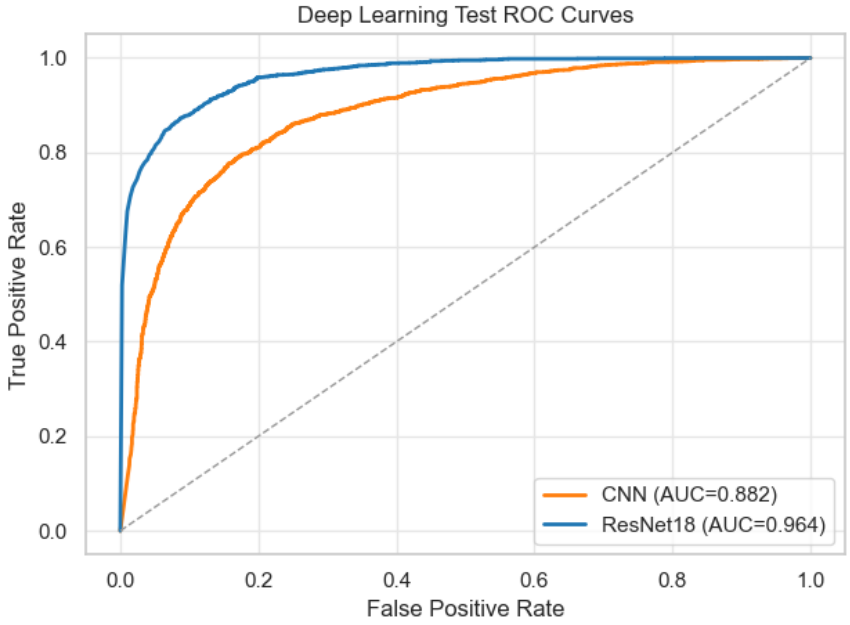
Table 5.3: Validation and test performance of deep learning models with validation-selected thresholds $\tau^* = 0.400$ (CNN) and $\tau^* = 0.425$ (ResNet18).

Model	Split	Accuracy	AUC	AP	Precision	Recall
CNN	Validation	0.983	0.998	0.998	0.981	0.983
ResNet18	Validation	1.000	1.000	1.000	1.000	1.000
CNN	Test	0.745	0.882	0.861	0.674	0.929
ResNet18	Test	0.528	0.964	0.963	0.508	1.000

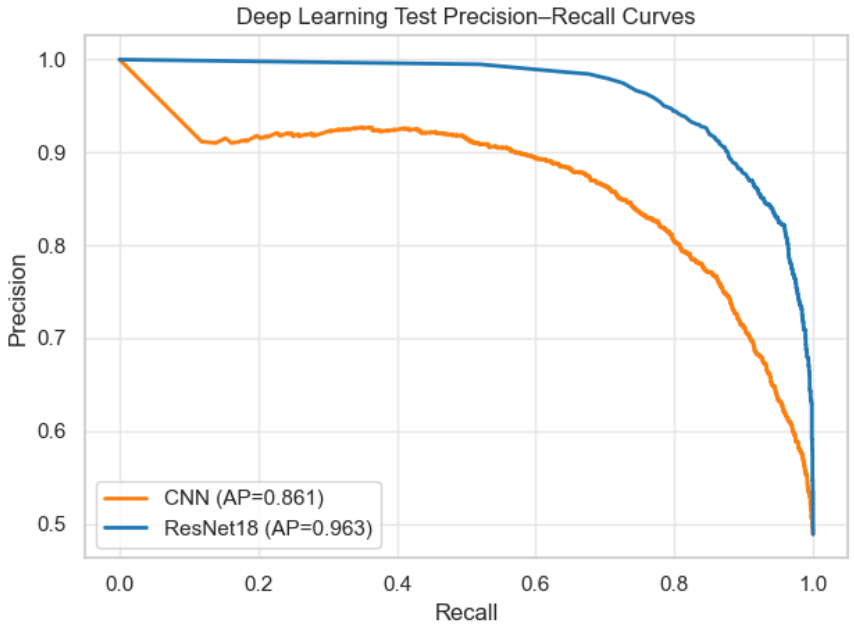
The results show both models achieving nearly perfect validation performance, confirming that the spectrogram-based representations are highly informative on the training distribution. On the test set, the CNN attains higher accuracy and precision under the validation-selected threshold τ^*, while ResNet18 achieves substantially higher AUC and AP. This pattern indicates that ResNet18 separates real and fake clips more effectively in terms of ranking, while the CNN transfers its validation threshold more successfully to the test distribution. The discrepancy between ResNet18's high AUC and AP and its low accuracy suggests that many deepfake clips are assigned high "real" probabilities under τ^*, an effect examined in more detail in the calibration and error analysis.

Figure 5.6a and Figure 5.6b visualize test performance across a range of decision thresholds. The ROC and precision–recall curves show that ResNet18 provides consistently stronger ranking performance than the CNN. The ResNet18 ROC curve remains closer to the top-left corner across most false positive rates, and its precision–recall curve maintains higher pre-

cision over a broad range of recall values. This is reflected in its higher test AUC and AP, indicating that ResNet18 orders real and fake clips more effectively across decision thresholds.



(a) Test ROC curves for CNN and ResNet18.



(b) Test precision–recall curves for CNN and ResNet18.

Figure 5.6: Test ROC and precision–recall curves for the CNN and ResNet18.

5.2.2 Calibration and Error Analysis

The gap between ResNet18's high test AUC and low test accuracy under the validation-selected threshold indicates a calibration problem. To investigate this, the distribution of predicted probabilities, the sensitivity of accuracy to the decision threshold, and the resulting error patterns were analyzed.

To analyze errors under the validation-selected threshold τ^*, confusion matrices are shown in Figure 5.7 and Figure 5.8. The CNN shows high recall for real audio and a moderate false positive rate for fake audio at $\tau^* = 0.400$. In contrast, ResNet18 at $\tau^* = 0.425$ predicts the real class for most test samples. This yields perfect recall for real audio but also labels many fake clips as real, which explains the lower test accuracy and precision at this threshold.

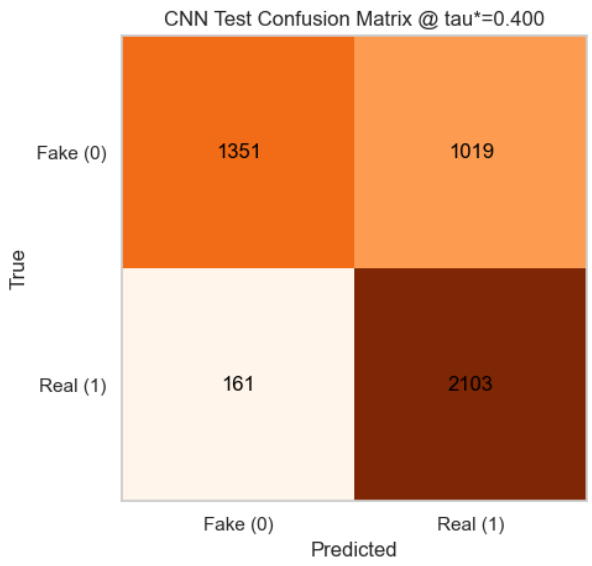


Figure 5.7: CNN Test Confusion Matrix.

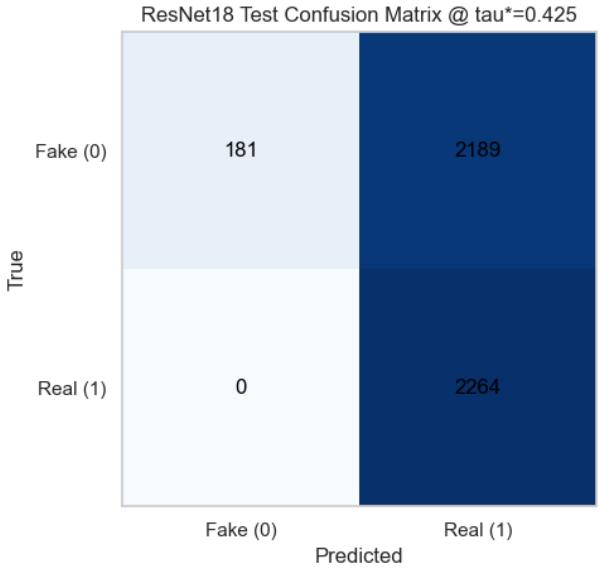


Figure 5.8: ResNet18 Test Confusion Matrix.

False positive samples were examined to understand why certain deepfakes were labeled as real under the validation-selected decision rule. Figure 5.9 shows the highest confidence false positive on the test set for ResNet18 under this setup. Despite being fake, the model assigns a probability of 1.0000 to the real class. The spectrogram contains a clear padded region from the fixed 4.0 second input window, yet the prediction remains highly confident. This highlights that threshold-based metrics depend on calibration and that some deepfakes

can produce confident errors even when structural cues are visible.

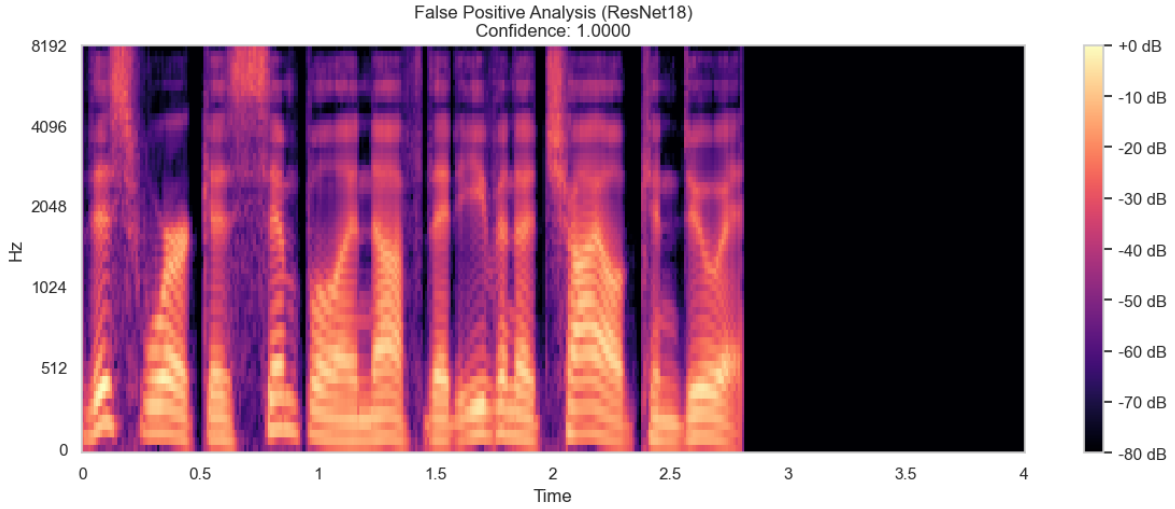
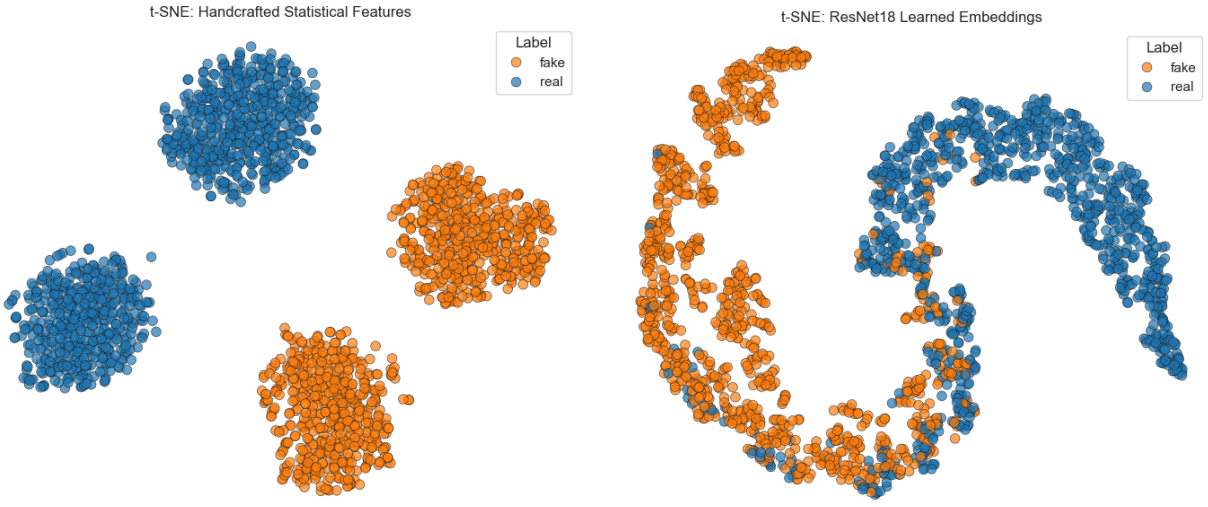


Figure 5.9: False positive example for ResNet18 on the test set under the validation-selected threshold framework.

To investigate the geometric structure of the learned feature spaces, t-Distributed Stochastic Neighbor Embedding (t-SNE) was employed to project the high-dimensional representations into two dimensions. Figure 5.10 presents a comparison between the handcrafted statistical features and the ResNet18 embeddings.



(a) Handcrafted Statistical Features. (b) ResNet18 Learned Embeddings.

Figure 5.10: t-SNE visualization of the test set ($N = 2000$).

Figure 5.10a shows several separated clusters with limited connectivity. This pattern suggests that the handcrafted features may be sensitive to hidden subgroups in the data instead of capturing a single real versus fake pattern. It also helps explain why the baseline models perform differently. The non-linear Random Forest can adapt to multiple disjoint clusters, while Logistic Regression is restricted to a single linear decision surface. As a result, it is less effective when the feature space contains several separated groups rather than a single dominant separation between real and fake examples. In contrast, the ResNet18 embeddings in Figure 5.10b show a smoother structure where the points form a continuous S-shaped curve. Real audio tends to appear near one end of the curve, while deepfake audio appears near the other end. This suggests that the network learned features that generalize better, which is consistent with its strong AUC results.

Figure 5.11 helps explain the performance discrepancy by showing a strongly polarized probability distribution. The model is highly overconfident, pushing most predictions toward values near 0 or 1 with very few intermediate probabilities. A substantial portion of fake samples in orange are assigned probabilities above 0.5, causing them to be labeled as real. As a result, the model makes confident errors, and threshold-based metrics become sensitive to calibration rather than ranking performance alone.

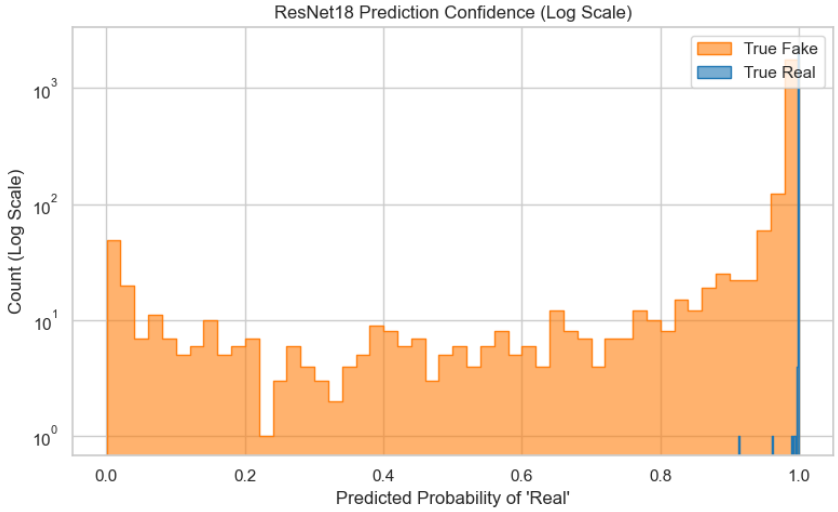


Figure 5.11: Probability histogram (log scale) for ResNet18.

This behavior is quantified in Figure 5.12, which plots test accuracy as a function of the decision threshold. Accuracy remains close to 0.53 over a wide range of thresholds $\tau < 0.9$, increasing only slightly before a sharp jump near $\tau \approx 0.99$, where accuracy peaks at 0.64. This indicates that the model still ranks samples well, consistent with its high AUC, but that the optimal decision boundary lies far to the right. The validation-selected threshold did not capture this shift, reflecting differences between the validation and test distributions.

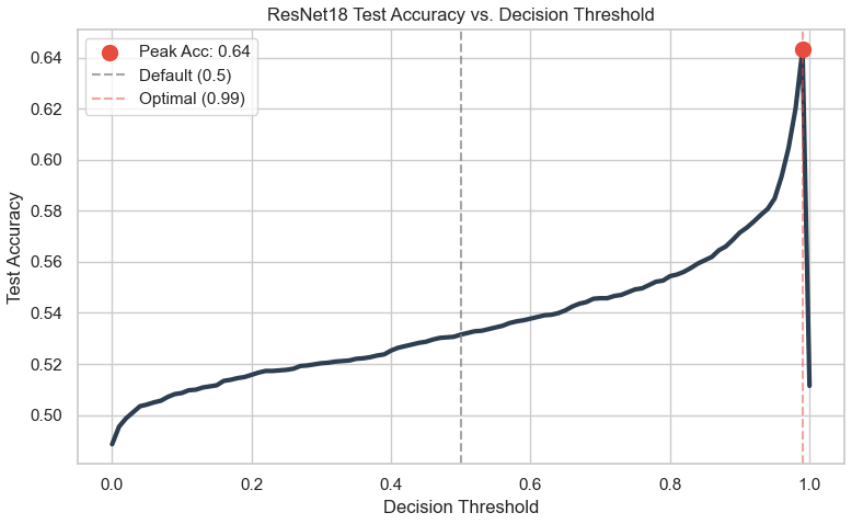


Figure 5.12: Accuracy vs. threshold for ResNet18.

To connect the metrics to acoustic patterns, a qualitative review examined samples with the highest and lowest model confidence. Figure 5.13 shows a 2×2 matrix of ResNet18 spectrograms. The top row contains correctly classified examples. The real clip displays clear harmonic structure and organized temporal patterns, while the fake clip appears smoother and less detailed in the higher frequency bands. The bottom row highlights the calibration issue. The bottom-left panel shows a deepfake classified as real with very high confidence, with $P \approx 1.0$. This clip visually resembles real speech and lacks obvious artifacts, which helps explain the error. The bottom-right panel shows the lowest-confidence real clip in the test set, yet its probability remains 0.9122, well above the 0.5 decision threshold. Together, these examples indicate that the low accuracy is driven primarily by false positives rather than false negatives.

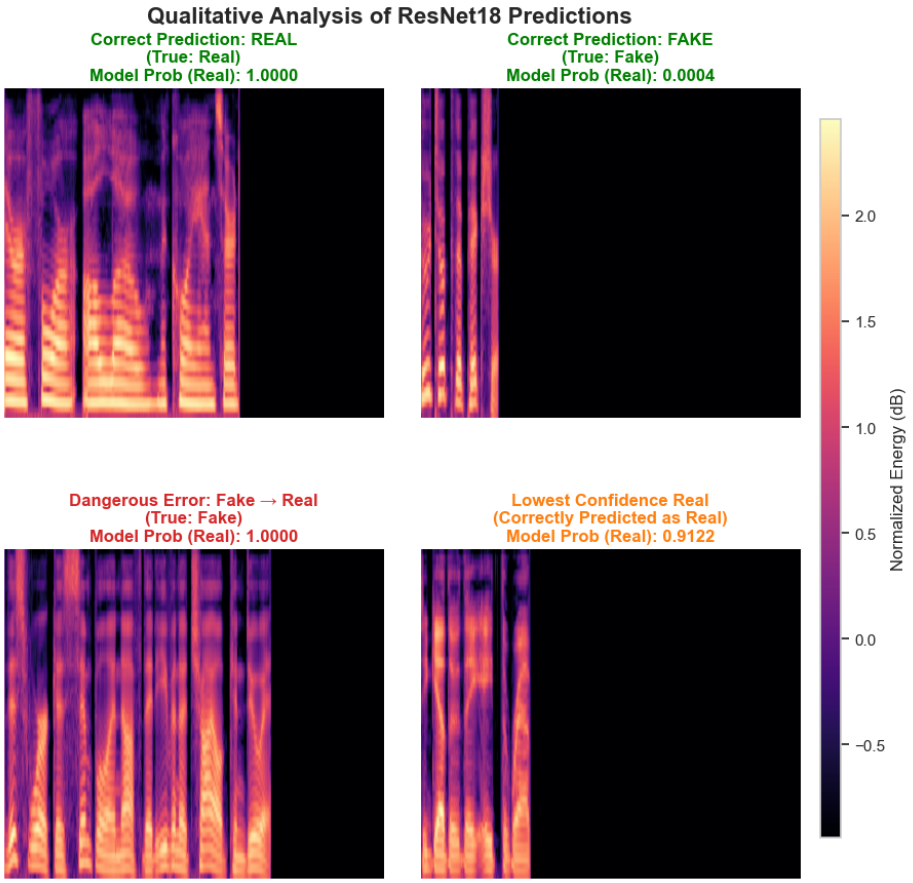


Figure 5.13: Spectrogram analysis of ResNet18 predictions.

5.2.3 Discussion

The results demonstrate that handcrafted features provide a strong baseline, while deep learning offers stronger ranking performance on this dataset. ResNet18 achieves higher test AUC and AP than the CNN, indicating better separation between real and fake across a range of decision thresholds. At the same time, under a decision threshold selected on validation and then fixed for test evaluation, the CNN achieves higher test accuracy than ResNet18. This shows that accuracy depends strongly on the chosen threshold and that strong ranking performance does not guarantee high accuracy at a single threshold.

The CNN results also illustrate the importance of training-time augmentation. SpecAugment reduces reliance on narrow time or frequency regions and encourages the model to learn

cues that are distributed across the spectrogram. This is consistent with the observation that deepfake artifacts are not always localized and that models can otherwise exploit dataset-specific shortcuts.

ResNet18 improves discrimination on the test set but exhibits a calibration and threshold transfer issue under the validation-selected decision rule. The test confusion matrix shows that at the selected threshold, ResNet18 predicts the real class for most samples, producing many fake clips incorrectly labeled as real. This explains the lower test precision and accuracy despite high AUC and AP. The false positive analysis also shows that some deepfake samples can look very similar to real speech in the spectrogram, which makes them difficult to reject at a fixed threshold. Taken together, these findings support reporting both threshold-free metrics such as AUC and AP and threshold-based metrics such as accuracy, precision, and recall. They also show that representation quality, calibration, and threshold selection all play critical roles in how deepfake models behave in practice.

CHAPTER 6

Concluding Remarks

6.1 Summary of Contributions

This thesis presented a comparative study of statistical versus deep learning methodologies for the detection of deepfake audio. A rigorous baseline was established using Logistic Regression and Random Forest models trained on handcrafted, non-learned features, with the Random Forest achieving a test accuracy of 0.815 and a test AUC of 0.888. Further analysis showed that deep learning performance depends strongly on training-time augmentation and on the decision threshold used for classification. Using a validation-selected decision threshold for threshold-based metrics, the CNN achieved a test accuracy of 0.745, while ResNet18 achieved a higher test AUC of 0.964.

Table 6.1 presents the final benchmark of all methodologies evaluated in this study. The results show a clear trade-off between threshold-based accuracy and threshold-free ranking performance.

Table 6.1: Final benchmark comparing statistical baselines and deep learning models.

Approach	**Model**	**Test Accuracy**	**Test AUC**
Statistical	Logistic Regression	0.609	0.652
Statistical	Random Forest	0.815	0.888
Deep Learning	CNN	0.745	0.882
Deep Learning	ResNet18	0.528	0.964

Taken together, the benchmark shows that learned representations can separate real and fake audio very effectively across a range of thresholds, as reflected in the strong test AUC of ResNet18, while tree-based statistical models remain competitive in terms of accuracy under a fixed decision rule. This indicates that classical signal processing features still capture important energy and spectral cues for deepfake detection, even when compared against modern deep learning methods.

The results also highlight that threshold-based metrics can differ sharply from threshold-free metrics. ResNet18 achieves strong test AUC and AP but attains low test accuracy at the validation-selected threshold, indicating a calibration and threshold transfer issue. For this reason, reporting both threshold-free metrics and threshold-based metrics is necessary for an accurate evaluation.

Finally, applying time and frequency masking through SpecAugment was found to support better generalization for the CNN training pipeline. Without this augmentation, models can rely on dataset-specific cues such as silence padding rather than learning distributed features. The false positive analysis further suggests that some deepfakes can exhibit highly coherent spectrogram structure, indicating that detection methods must evolve alongside improvements in audio generation.

6.2 Computational Efficiency and Memory

Beyond classification performance, the practical utility of a deepfake detection system depends on its computational efficiency. Inference latency and memory footprint were benchmarked for the Random Forest and the ResNet18 model to evaluate their suitability for real-time deployment.

Table 6.2 summarizes the results. Inference latency represents the average time required to process a single 4-second audio clip on the test environment.

Table 6.2: Computational cost comparison for Random Forest and ResNet18.

Model	**Model Size**	**Inference Time**	**Throughput**
Random Forest	130.06 MB	0.013 ms	∼76,154 samples/s
ResNet18	42.65 MB	9.30 ms	∼108 samples/s

The comparison reveals a distinct trade-off. The Random Forest is approximately 700 times faster than the ResNet18 model, capable of processing over 76,000 clips per second. This makes it a strong choice for high-volume screening or real-time monitoring when latency is the main constraint. However, this speed comes at the cost of memory efficiency. The Random Forest model size is roughly three times larger than that of ResNet18. This occurs because the ensemble stores the structure of hundreds of decision trees, while ResNet18 has a fixed number of parameters set by its architecture. Together, these results support a staged deployment strategy. The Random Forest can act as a fast first-pass filter, while ResNet18 is reserved for more detailed analysis of flagged audio.

6.3 Limitations

Despite the progress achieved in this thesis, several limitations remain. A primary concern is dataset bias from using only the Fake-or-Real dataset. Although the test set was held out, it likely shares recording conditions such as microphone characteristics and compression artifacts with the training data. This suggests that performance may degrade when applied to audio from uncontrolled environments such as social media. In addition, the models were trained on synthesis and manipulation methods represented in this dataset, and generalization to newer generation pipelines such as diffusion-based methods or modern voice conversion systems remains an open question. Finally, the deep learning results show that threshold-based accuracy can be sensitive to the decision threshold and may not transfer cleanly across splits, which highlights calibration as an additional limitation for deployment. The scope of models and hyperparameter search was also limited to a small set of archi-

tectures and training settings, so stronger configurations may be possible. Moreover, all experiments were conducted on a single curated dataset without human listener studies or live deployment tests, leaving real-world performance and usability as open questions.

6.4 Future Work

Future research should focus on improving generalization and making threshold-based decisions more reliable. A clear next step is cross-dataset evaluation. Testing the models on external benchmarks such as ASVspoof or on in-the-wild datasets would provide a stronger assessment of generalization under distribution shift. Another direction is improving calibration and threshold selection. Methods such as temperature scaling or validation-based calibration procedures could reduce overconfidence and yield more reliable threshold-based performance. Additional gains may be possible through stronger augmentation and training strategies, including adversarial training that encourages invariance to speaker identity and recording conditions. Finally, incorporating interpretability tools such as Grad-CAM can help identify which time and frequency regions drive predictions and can verify that the models rely on manipulation cues rather than dataset artifacts.

6.5 Final Remarks

This research shows that deep learning provides stronger ranking performance, while statistical baselines remain competitive in accuracy and offer computational simplicity. The results suggest that future systems may benefit from combining deep representations with lightweight decision rules and calibration procedures to achieve effective separation between real and fake audio and more reliable performance at fixed thresholds.

References

[1] Moustafa Alzantot, Ziqi Wang, and Mani B Srivastava. Deep residual neural networks for audio spoofing detection. In *Proc. Interspeech 2019*, pages 1078–1082, 2019.

[2] Ameer Hamza, Abdul Rehman Javed, Farkhund Iqbal, Natalia Kryvinska, and Ahmad S Almadhor. Deepfake audio detection via MFCC features using machine learning. *IEEE Access*, 10:134018–134028, 2022.

[3] Zahra Khanjani, Gabrielle Watson, and Vandana P Janeja. Audio deepfakes: A survey. *Frontiers in Big Data*, 5:1001063, 2023.

[4] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. SpecAugment: A simple data augmentation method for automatic speech recognition. In *Proc. Interspeech 2019*, pages 2613–2617, 2019.

[5] Massimiliano Todisco, Xin Wang, Ville Vestman, Md Sahidullah, Hector Delgado, Andreas Nautsch, Junichi Yamagishi, Nicholas Evans, Tomi Kinnunen, and Kong Aik Lee. ASVspoof 2019: Future horizons in spoofed and fake audio detection. In *Proc. Interspeech 2019*, pages 1008–1012, 2019.

[6] Xin Wang and Junichi Yamagishi. A comparative study on recent neural spoofing countermeasures for synthetic speech detection. In *Proc. Interspeech 2021*, pages 4259–4263, 2021.

[7] Jiangyan Yi, Chenglong Wang, Jianhua Tao, Xiaohui Zhang, and Yan Zhao. Audio deepfake detection: A survey. *arXiv preprint arXiv:2308.14970*, 2023.