Voice-Based Speaker Recognition Using MFCC and LSTM

Team Members

Abbas Ali, Shravya Kumbham, Hesham Aldahbali

Department of Electrical and Computer Engineering
University of Michigan - Dearborn
Dearborn, Michigan, USA

ECE 5831 - Pattern Recognition and Neural Networks

December 2025

Contents

Abstract

This final report presents a comprehensive study on voice-based speaker recognition using Mel-Frequency Cepstral Coefficients (MFCC) and Long Short-Term Memory (LSTM) networks. Through a systematic four-phase approach, we developed and validated a speaker identification system achieving 95-98% accuracy across multiple datasets including Kaggle public figures, YouTube celebrities, and custom recordings from team members.

Most significantly, we discovered and documented a critical device bias phenomenon where models trained on speakers using different recording devices learned to identify microphone signatures rather than voice characteristics. This unexpected finding emerged during Phase 3 when our model achieved high training accuracy ($> 95\%$) but failed completely on new test recordings. Through systematic debugging—including fundamental frequency analysis, silence removal, and rolling window techniques—we ultimately designed a cross-device experiment that revealed the root cause: the model had learned to distinguish between recording devices instead of human voices.

This discovery led to the development of a controlled data collection methodology in Phase 4, where all speakers used identical recording equipment during training. This simple yet crucial modification forced the model to learn actual voice features (pitch patterns, formant frequencies, vocal tract resonance) instead of device signatures. Validation testing confirmed that the Phase 4 model correctly identified speakers regardless of the recording device used during testing, achieving 95.74% validation accuracy and successful cross-device generalization.

Our work demonstrates that data collection methodology is as critical as model architecture in machine learning applications. The device bias phenomenon—where perfectly correlated confounding variables (device type) dominate model learning over intended features (voice characteristics)—has broad implications for ML deployment in biometric authentication, fairness in AI systems, and educational practices in data science.

1 Introduction

1.1 Background and Motivation

Speaker recognition represents a fundamental challenge in biometric authentication and human-computer interaction, with applications spanning security systems, personalized voice assistants, forensic analysis, and telecommunications. The core problem involves extracting discriminative features from audio signals that uniquely identify individual speakers while remaining robust to variations in recording conditions, background noise, speaking style, and hardware characteristics.

Our project was motivated by dual objectives: first, to gain hands-on experience implementing a complete speaker recognition pipeline from feature extraction through model deployment; and second, to understand the practical challenges encountered during real-world deployment that often remain undocumented in academic literature. While existing research extensively covers algorithmic improvements and theoretical foundations, fewer studies systematically examine how data collection methodology impacts what models actually learn—a critical gap we aimed to address through empirical investigation.

1.2 Why This Project is Interesting

This project is particularly interesting for several reasons:

1. Real-World Relevance: Voice-based authentication is increasingly deployed in consumer devices, banking applications, and smart home systems. Understanding both its capabilities and limitations has direct practical implications for millions of users.

2. The Unexpected Discovery: What began as a straightforward implementation project evolved into a scientific investigation when our custom dataset produced puzzling results. The journey from confusion to discovery—systematically testing hypotheses until uncovering the device bias phenomenon—mirrors real scientific inquiry and demonstrates how unexpected findings often yield the most valuable insights.

3. Bridging Theory and Practice: Academic papers typically report successful results on carefully curated datasets. Our project reveals the messy reality of ML deployment: high validation accuracy doesn't guarantee learning the intended features, and hidden correlations in training data can lead models to exploit spurious patterns. This gap between laboratory performance and real-world behavior is rarely documented but critically important.

4. Broad Implications: The device bias phenomenon extends beyond speaker recognition. Any ML system trained on data where confounding variables (camera type in facial recognition, scanner model in medical imaging, sensor brand in IoT applications) correlate with target labels faces similar risks. Our findings contribute to understanding dataset bias and spurious correlations—topics of growing importance in ML fairness and robustness research.

5. Educational Value: This project provides a concrete case study for teaching data collection best practices. The narrative arc—baseline validation, successful results, unexpected failure, systematic debugging, root cause discovery, and validated solution—demonstrates the scientific method applied to machine learning.

1.3 Project Evolution

Our team's journey to this project topic reflects the iterative nature of research planning. Initially, we proposed implementing image compression using autoencoders, motivated by interest in generative models and dimensionality reduction. However, after consultation with course instructors, we recognized that optimizing and validating such a system within one semester would require computational resources (GPU hours, large image datasets) beyond our available infrastructure.

We pivoted to voice-based speaker recognition, which offered several advantages:

- **Manageable Computational Requirements:** Audio files are smaller than images, and 1-second chunks require less memory than high-resolution image batches.
- **Clear Evaluation Metrics:** Speaker identification is a straightforward classification task with interpretable accuracy metrics.
- **Practical Relevance:** Voice authentication is widely deployed, making our work directly applicable.
- **Custom Dataset Creation:** We could easily record our own voices, enabling investigation of real-world deployment scenarios.
- **Established Techniques:** MFCC feature extraction and LSTM modeling are well-documented, allowing us to focus on implementation and discovery rather than algorithmic innovation.

This pivot proved fortuitous, as creating custom datasets ultimately led to our most significant finding: the device bias phenomenon.

1.4 Key Contributions

This work makes four primary contributions to the understanding of speaker recognition systems:

1. Systematic Implementation and Validation: We provide complete documentation of MFCC+LSTM speaker recognition across three distinct datasets (Kaggle public figures, YouTube celebrities, custom team recordings), demonstrating the approach's effectiveness and establishing performance baselines.

2. Device Bias Discovery and Documentation: We identified, rigorously validated, and thoroughly documented a device signature learning phenomenon where models trained with device-speaker correlation learn to classify recording devices rather than human voices. This finding emerged through systematic scientific investigation including:

- Fundamental frequency analysis
- Silence detection and removal
- Rolling window temporal analysis
- Cross-device validation experiments

3. Methodological Solution with Validation: We developed and validated a controlled data collection protocol that eliminates device bias by ensuring all speakers use identical recording equipment. Cross-device testing confirmed this approach forces models to learn voice characteristics that generalize across hardware.

4. Practical ML Insights: We documented critical lessons about the gap between validation accuracy and real-world performance, the importance of controlling confounding variables in training data, and the necessity of deployment-realistic testing conditions.

1.5 Document Organization

The remainder of this report is organized as follows:

Section 2 provides comprehensive literature review covering MFCC feature extraction fundamentals, traditional and deep learning approaches to speaker recognition, relevant datasets, and computational tools.

Section 3 details our methodology including system architecture, MFCC configuration, LSTM model design, and evaluation protocols.

Section 4 presents results from all four project phases: Kaggle baseline validation, YouTube dataset testing, device bias discovery with the flawed custom dataset, and successful same-device solution.

Section 5 discusses key findings, implications for ML practice, limitations of our study, and future research directions.

Section 6 concludes with a summary of contributions and broader impact.

2 Related Work and Literature Review

2.1 Speech Feature Extraction: MFCC Fundamentals

Mel-Frequency Cepstral Coefficients (MFCC) have been the dominant feature representation for speech and speaker recognition since their introduction by Davis and Mermelstein in 1980 [1]. Their seminal work demonstrated that features modeling the human auditory system's non-linear frequency perception significantly outperform linear spectral features for speech recognition tasks.

MFCCs capture the spectral envelope of speech signals through a multi-stage process:

1. **Frame Segmentation:** Divide audio into overlapping short-time frames (typically 20-40ms) to capture quasi-stationary spectral properties.
2. **Windowing:** Apply a window function (Hamming or Hann) to reduce spectral leakage from frame boundaries.
3. **FFT:** Compute the power spectrum via Fast Fourier Transform.
4. **Mel Filterbank:** Apply triangular filters spaced along the mel scale, which approximates human pitch perception (linear below 1kHz, logarithmic above).
5. **Logarithm:** Compute log-energy in each mel band, modeling loudness perception.
6. **DCT:** Apply Discrete Cosine Transform to decorrelate filterbank energies and compress information.

The resulting MFCC coefficients effectively represent vocal tract shape, which is speaker-specific and relatively stable across different utterances. Rabiner and Juang's comprehensive textbook [4] provides detailed mathematical foundations for speech signal processing, explaining why cepstral analysis successfully separates source (vocal cord vibration) from filter (vocal tract resonance) characteristics.

In our implementation, we extract 13 MFCC coefficients per frame, following standard practice that balances information capture with computational efficiency. The choice of 13 coefficients has been empirically validated across decades of speech research as capturing sufficient spectral detail without redundant information.

2.2 Traditional Speaker Recognition Approaches

Before the deep learning revolution, speaker recognition systems relied primarily on statistical modeling techniques. Reynolds and Rose [3] demonstrated that Gaussian Mixture Models (GMMs) could achieve robust text-independent speaker identification by modeling each speaker's acoustic space as a mixture of multivariate Gaussian distributions. Their Universal Background Model (UBM) approach—training a speaker-independent model on diverse speech data, then adapting it to individual speakers—became the dominant paradigm for over a decade.

Campbell's comprehensive tutorial [5] covers both speaker verification (one-to-one matching: "Is this person who they claim to be?") and speaker identification (one-to-many matching: "Who is speaking?"). His work emphasizes critical distinctions between text-dependent systems (requiring specific phrases) and text-independent systems (working with any speech), as well as the importance of score normalization techniques to handle variability in recording conditions.

Bimbot et al. [6] provide an extensive review of text-independent speaker verification, discussing various modeling approaches including GMM-UBM, support vector machines with supervectors, and joint factor analysis. Their work highlights challenges still relevant today: channel compensation (accounting for microphone and transmission effects), session variability (differences in recording conditions), and aging effects (voice changes over time). Notably, their discussion of channel compensation relates directly to our device bias findings—traditional systems explicitly modeled and compensated for channel effects, while deep learning systems must either learn invariance or have it enforced through data collection.

2.3 Deep Learning Revolution in Speaker Recognition

The application of deep learning to speech and speaker recognition has transformed the field over the past decade. Hinton et al.'s landmark paper [8] demonstrated that deep neural networks significantly outperform GMM-based systems for acoustic modeling in speech recognition, achieving dramatic error rate reductions. Their work established that deep architectures could learn hierarchical representations—lower layers capturing phonetic features, higher layers capturing linguistic patterns—more effectively than handengineered features.

For speaker recognition specifically, the temporal nature of speech signals motivates recurrent architectures. Graves et al. [9] showed that recurrent neural networks, particularly Long Short-Term Memory (LSTM) networks, excel at modeling temporal dependencies in speech. Their work on speech recognition demonstrated that LSTMs could capture long-range acoustic patterns spanning multiple frames, phonemes, or even words.

Hochreiter and Schmidhuber's original LSTM paper [2] introduced the architecture's key innovation: gated memory cells that can maintain information over many time steps, addressing the vanishing gradient problem that plagued earlier recurrent networks. The LSTM's ability to learn what information to store, when to forget old information, and when to output results makes it particularly suitable for variable-length speech sequences where relevant patterns may occur at different time scales.

Modern speaker recognition increasingly employs end-to-end learning approaches. Snyder et al. [10] introduced x-vectors, extracting fixed-dimensional speaker embeddings from time-delay neural networks (TDNNs) trained directly on speaker classification. Their approach achieves state-of-the-art performance by learning representations optimized for speaker discrimination rather than relying on hand-crafted features. Importantly, x-vectors trained on large-scale data generalize well to new speakers and recording conditions—a capability we aimed to replicate on a smaller scale.

Li et al.'s DeepSpeaker system [12] demonstrates end-to-end training for speaker verification, learning embeddings that maximize between-speaker distances while minimizing within-speaker variability. Their work emphasizes data augmentation (adding noise, reverberation, codec distortions) to improve robustness—techniques we did not employ but which could potentially mitigate device bias.

Variani et al. [13] showed that deep neural networks can achieve strong speaker verification performance even in small-footprint scenarios suitable for mobile deployment. Their work is particularly relevant to our project, which involves mobile device recordings, demonstrating that compact models can still learn discriminative speaker representations.

2.4 Datasets and Benchmarks

Large-scale datasets have driven recent progress in speaker recognition. Nagrani et al.'s VoxCeleb dataset [11] contains over 100,000 utterances from 1,251 celebrities extracted from YouTube videos, providing diverse acoustic conditions, languages, and speaking styles. VoxCeleb's scale enables training robust deep learning models, but its construction methodology (collecting YouTube videos) may inadvertently introduce device correlations we discovered—different celebrities might consistently use similar recording setups.

While we did not use VoxCeleb due to computational constraints, its existence validates our approach of collecting YouTube data (Phase 2) for testing on real-world noisy audio rather than only clean laboratory recordings.

2.5 Tools and Frameworks

Our implementation leverages two key software libraries:

Librosa [14] provides efficient Python implementations of audio signal processing operations including MFCC extraction, spectral analysis, and time-frequency representations. Its design prioritizes both correctness (carefully validated against reference implementations) and usability (intuitive API), making it the de facto standard for audio processing in Python-based ML research.

TensorFlow [15] offers a flexible framework for deep learning model development, training, and deployment. Its Keras API provides high-level abstractions for building neural networks (Sequential models, functional API) while maintaining access to low-level operations when needed. TensorFlow's automatic differentiation, GPU acceleration, and production deployment tools made it ideal for our project.

2.6 Gap in Existing Literature

While extensive research addresses algorithmic improvements—better neural architectures, novel loss functions, advanced embedding techniques—fewer studies systematically examine how data collection methodology impacts what models learn in practice. Most papers present results on carefully curated benchmark datasets (VoxCeleb, TIMIT, NIST SRE) where data collection followed rigorous protocols.

Our work contributes empirical evidence that confounding variables in training data can lead models to learn unintended features even when achieving high validation accuracy. The device bias phenomenon we discovered—where device-speaker correlation causes models to exploit microphone signatures—represents a practical deployment consideration rarely discussed in academic literature.

This gap between laboratory conditions and real-world deployment challenges has been noted in broader ML fairness and robustness research, where dataset bias, spurious correlations, and distribution shift are active research areas. Our findings extend these concepts specifically to speaker recognition, providing a concrete case study of how uncontrolled variables in data collection can undermine system performance despite strong validation metrics.

3 Methodology

3.1 System Architecture Overview

Our speaker recognition system implements a standard pipeline consisting of four stages: audio preprocessing, feature extraction, model training, and evaluation. This modular design allows systematic validation at each stage and facilitates debugging when unexpected results occur (as indeed happened in Phase 3).

3.2 Audio Preprocessing

Raw audio files undergo several preprocessing steps to ensure consistency and compatibility with feature extraction:

1. Resampling: All audio is resampled to 16 kHz, the standard sampling rate for speech processing. This rate captures frequencies up to 8 kHz (Nyquist theorem), sufficient for human speech which has most energy below 4 kHz. Lower rates reduce computational requirements while retaining discriminative information.

2. Channel Conversion: Stereo recordings are converted to mono by averaging left and right channels. Since speaker characteristics are present in both channels, this simplification reduces data dimensionality without information loss.

3. Segmentation: Long audio recordings are divided into 1-second chunks (16,000 samples each). This fixed duration enables batched processing and creates consistent input dimensions for the neural network. One second provides enough temporal context for MFCC patterns while remaining short enough to assume quasi-stationarity.

4. Amplitude Normalization: Each chunk's amplitude is normalized to $[-1, 1]$ range to account for volume variations across recordings. This ensures consistent signal levels regardless of recording gain settings.

3.3 Feature Extraction: MFCC Configuration

For each 1-second audio chunk, we extract Mel-Frequency Cepstral Coefficients following this process:

$$\text{MFCC}(t) = \text{DCT}\left(\log\left(M \cdot |F(x(t))|^2\right)\right) \tag{1}$$

where:

- $x(t)$ is the time-domain signal
- F denotes Short-Time Fourier Transform (STFT)
- M represents the mel-scale filterbank matrix (26 filters)
- DCT is Discrete Cosine Transform

Our specific configuration parameters:

- **Number of Coefficients:** 13 MFCC coefficients (excluding C_0, the energy term)
- **Time Frames:** 32 frames per second of audio
- **Window Length:** 512 samples (32ms) with 50% overlap
- **Feature Matrix:** (32×13) per 1-second sample

The choice of 13 coefficients follows decades of empirical validation in speech recognition research. Higher-order coefficients primarily capture speaker-independent phonetic information rather than speaker-specific characteristics.

After extraction, features undergo z-score normalization to zero mean and unit variance:

$$\text{MFCC}_{\text{norm}} = \frac{\text{MFCC} - \mu}{\sigma} \tag{2}$$

This normalization improves gradient-based optimization convergence during neural network training.

3.4 LSTM Model Architecture

Our neural network architecture consists of four layers designed to learn temporal patterns in MFCC sequences:

Layer 1 - Input: Receives $(32, 13)$ MFCC feature matrices representing 32 time frames with 13 coefficients each.

Layer 2 - LSTM: 128 hidden units process the temporal sequence, learning dependencies across the 32 frames. The LSTM's gated architecture (forget gate, input gate, output gate) enables learning both short-term patterns (phoneme transitions) and longer-term patterns (prosodic characteristics) relevant to speaker identification.

Layer 3 - Dense (Hidden): 64 units with ReLU activation. This fully connected layer learns non-linear combinations of LSTM outputs, creating higher-level speaker-discriminative representations.

Layer 4 - Output: N units (where N equals number of speakers) with Softmax activation, producing probability distributions over speaker identities.

Training Configuration:

- **Loss Function:** Sparse categorical crossentropy (appropriate for mutually exclusive classes)
- **Optimizer:** Adam with default learning rate (0.001)
- **Batch Size:** 32 samples
- **Early Stopping:** Monitors validation loss with patience of 3 epochs, restoring best weights

The early stopping mechanism prevents overfitting by halting training when validation performance stops improving, ensuring the model generalizes rather than memorizing training examples.

3.5 Dataset Splitting Strategy

For all experiments, we employ stratified random splitting to maintain class balance across subsets:

- **Training Set:** 70% of data for model parameter optimization
- **Validation Set:** 15% for hyperparameter tuning and early stopping decisions
- **Test Set:** 15% held out completely for final performance evaluation

Stratification ensures each subset contains proportional representation from all speakers, preventing class imbalance that could bias evaluation metrics.

3.6 Evaluation Metrics

We assess model performance using multiple complementary metrics:

1. Classification Accuracy: Percentage of correctly identified speakers, providing an overall performance measure.

2. Confusion Matrix: Per-speaker accuracy breakdown revealing systematic errors (e.g., consistent confusion between specific speaker pairs).

3. Training Curves: Accuracy and loss plotted over epochs, indicating learning progression, convergence, and potential overfitting.

4. Cross-Device Testing: For Phase 4, testing with recordings from devices not used during training validates that the model learned voice characteristics rather than device signatures.

This multi-faceted evaluation proved crucial—Phase 3 achieved high accuracy and good-looking training curves but failed cross-device testing, revealing that validation metrics alone can be misleading when training data contains hidden correlations.

4 Results

We present results from four distinct experimental phases, each designed to progressively validate our approach and, unexpectedly, to discover the device bias phenomenon.

4.1 Phase 1: Kaggle Baseline Dataset

4.1.1 Dataset Description and Rationale

To validate our MFCC+LSTM implementation before collecting custom data, we began with a publicly available Kaggle dataset containing pre-segmented 1-second audio clips from five prominent public figures:

- Nelson Mandela (1,500 samples)
- Benjamin Netanyahu (1,500 samples)
- Jens Stoltenberg (1,500 samples)
- Julia Gillard (1,501 samples)
- Margaret Thatcher (1,500 samples)

Total: 7,501 samples from professional recordings with consistent audio quality, minimal background noise, and clean speech.

This dataset offered several advantages for baseline validation:

1. Pre-segmented into 1-second chunks, eliminating segmentation errors

2. Professional recording quality ensures features reflect voice characteristics
3. Diverse speakers (different genders, ages, accents) test model robustness
4. Established benchmark for comparing our implementation to prior work

4.1.2 Training Process and Results

After extracting MFCC features and creating train/validation/test splits (70/15/15%), we trained the LSTM model with early stopping. Table 1 summarizes performance:

Table 1: Phase 1: Kaggle Dataset Performance Metrics

Metric	**Value**
Training Accuracy	98.44%
Validation Accuracy	97.07%
Test Accuracy	97.69%
Epochs Trained	9 (stopped early)
Training Time	∼8 minutes

The model achieved 97.69% test accuracy, demonstrating that MFCC features combined with LSTM architecture effectively distinguish between speakers on clean audio. Confusion matrix analysis revealed minimal inter-speaker errors, with most mistakes occurring between Netanyahu and Stoltenberg—both male speakers with similar age and vocal characteristics.

Per-Speaker Accuracy:

- Nelson Mandela: 99.3% (low, distinctive voice)
- Benjamin Netanyahu: 96.2% (some confusion with Stoltenberg)
- Jens Stoltenberg: 95.8% (some confusion with Netanyahu)
- Julia Gillard: 98.9% (distinctive female voice)
- Margaret Thatcher: 99.1% (distinctive accent and vocal quality)

Training curves showed smooth convergence without overfitting—validation accuracy closely tracked training accuracy throughout, and early stopping triggered at epoch 9 when validation loss stopped decreasing.

4.1.3 Key Findings from Phase 1

This baseline phase validated several critical assumptions:

1. **MFCC features are discriminative:** 13 coefficients across 32 time frames capture sufficient speaker-specific information for high-accuracy classification.
2. **LSTM architecture is effective:** The 128-unit LSTM layer successfully learns temporal patterns distinguishing speakers, confirming that recurrent processing of sequential MFCC frames adds value beyond frame-independent classification.
3. **Early stopping prevents overfitting:** The modest gap between training (98.44%) and test (97.69%) accuracy indicates good generalization.
4. **Baseline performance established:** 97.69% accuracy on clean audio provides a reference point for subsequent phases with noisier, real-world data.

Encouraged by these results, we proceeded to Phase 2 to test robustness on more challenging, realistic audio conditions.

4.2 Phase 2: YouTube Public Figures Dataset

4.2.1 Motivation and Dataset Description

While Phase 1 validated our technical approach, professional studio recordings don't represent typical deployment scenarios. Real-world speaker recognition must handle:

- Background noise (crowds, music, traffic)
- Reverberation and room acoustics
- Compression artifacts (MP3, streaming codecs)
- Varying microphone quality
- Non-stationary noise sources

To test robustness under these conditions, we collected audio from YouTube videos featuring three contemporary public figures:

- Donald Trump (1,649 samples) - political rallies, speeches, interviews
- Kamala Harris (1,123 samples) - political speeches, interviews

- Elon Musk (2,353 samples) - tech presentations, interviews, podcasts

Total: 5,125 samples with realistic noise, varied recording setups, and diverse acoustic conditions.

4.2.2 Data Collection and Preprocessing Challenges

Collecting this dataset required additional steps beyond Phase 1:

1. Download speech videos from YouTube using `youtube-dl`
2. Extract audio tracks (typically MP3 format)
3. Convert to WAV (uncompressed) format
4. Segment into 1-second chunks using PyDub library
5. Manual quality control:
    - Remove chunks with only applause or music
    - Discard chunks with overlapping speech from multiple speakers
    - Filter out silence-only chunks

This manual curation reduced raw data by approximately 20%, ensuring training examples contained actual speech from the target speaker.

4.2.3 Training Results and Analysis

Despite noisier conditions, the model achieved remarkably high performance (Table 2):

Table 2: Phase 2: YouTube Dataset Performance Metrics

Metric	**Value**
Training Accuracy	98.77%
Validation Accuracy	99.09%
Test Accuracy	98.70%
Epochs Trained	14 (stopped early)
Training Time	∼12 minutes

Notably, test accuracy (98.70%) actually *exceeded* Phase 1 (97.69%), defying initial expectations that noisy data would degrade performance. This counterintuitive result likely stems from:

1. Larger acoustic variability forcing the model to learn robust features
2. More diverse speaking styles (impromptu vs. scripted speech) providing richer training signal
3. Modern audio quality in YouTube content (though compressed) being quite good

Per-Speaker Performance:

- Trump: 100% accuracy (353/353 correct) - highly distinctive speech patterns
- Kamala Harris: 96.37% (239/248 correct) - occasional confusion with Trump
- Elon Musk: 99.40% (167/168 correct) - distinctive South African accent

The confusion between Trump and Kamala Harris is interesting—both are American politicians often using similar rhetorical styles, prosody, and emphatic speech patterns, suggesting the model sometimes focuses on speaking style in addition to intrinsic vocal characteristics.

4.2.4 Key Findings from Phase 2

Phase 2 provided crucial validation for real-world deployment:

1. **Background noise robustness:** The MFCC+LSTM approach handles realistic noise well, with background sounds, applause, and room reverberation not significantly degrading accuracy.
2. **Compression resilience:** YouTube audio compression (typically 128-256 kbps AAC/Opus) does not remove discriminative speaker information, suggesting the system could work with streamed or transmitted audio.
3. **Cross-domain validation:** Success on both professional recordings (Phase 1) and amateur YouTube content (Phase 2) indicates the approach generalizes across recording conditions.
4. **Strong baseline established:** 98.70% accuracy became our benchmark for evaluating the custom dataset in Phase 3.

With confidence in our implementation's technical soundness and real-world robustness, we moved to Phase 3: creating a custom dataset with team member voices to demonstrate the system on a completely new set of speakers.

4.3 Phase 3: Custom Dataset and the Device Bias Discovery

4.3.1 Motivation and Dataset Creation

Having validated our approach on public datasets, we sought to demonstrate the system's effectiveness on completely new speakers—our team members. This phase would answer the critical question: "Can we build a working speaker recognition system from scratch with our own voices?"

We collected custom training data using a simple protocol:

- Each team member recorded 20 minutes of speech using their own mobile phone
- Recording content: reading articles, casual conversation, project discussions
- Each 20-minute file was automatically segmented into 1-second chunks using PyDub
- Generated approximately 1,200 samples per speaker
- Total dataset: 3,600 samples (Abbas: 1,200, Hesham: 1,200, Shravya: 1,200)

This automated pipeline (long recording → automatic chunking) simplified data collection compared to manually recording individual 1-second samples.

4.3.2 Initial Training Results

Training proceeded smoothly with encouraging metrics (Table 3):

Table 3: Phase 3: Initial Custom Dataset Performance

Metric	**Value**
Training Accuracy	96.12%
Validation Accuracy	95.33%
Test Accuracy	94.87%
Epochs Trained	12 (stopped early)

The training curves looked perfect—smooth convergence, no overfitting, validation accuracy tracking training accuracy closely. The confusion matrix showed good separation between speakers with minimal inter-speaker errors. By all standard metrics, this model appeared successful.

4.3.3 The Failure: Real-World Testing

Confident in our results, we tested the model on new 2-minute recordings from each team member captured days later. The results were shocking:

Prediction Results on New Test Audio:

- Abbas's new recording: Predicted as **Abbas** (correct)
- Hesham's new recording: Predicted as **Hesham** (correct)
- Shravya's new recording: Predicted as **Abbas** (WRONG!)

The model consistently misidentified Shravya as Abbas with high confidence (¿95%). This complete failure contradicted the 95% validation accuracy. Something was fundamentally wrong.

4.3.4 Systematic Troubleshooting Journey

We embarked on extensive troubleshooting to understand why the model failed on new audio despite high training accuracy.

Hypothesis 1: Fundamental Frequency Overlap

Theory: Team members might have similar fundamental frequencies (pitch), causing confusion.

Action: Plotted spectrograms for each speaker and analyzed base frequency ranges and formant structures.

Result: Base frequencies were DISTINCT across all three speakers—this was NOT the issue. Spectrograms revealed clear differences in pitch ranges.

Hypothesis 2: Silence in Audio Chunks

Theory: Some 1-second chunks might contain only silence or pauses, providing no useful voice information.

Action: Implemented energy-based silence detection and removal, filtering out chunks below an energy threshold. Retrained the model on cleaned data.

Result: NO improvement in predictions. The model still misidentified Shravya's new recordings as Abbas.

Hypothesis 3: Fixed Chunking Misses Features

Theory: Fixed 0-1s, 1-2s chunking might miss important voice patterns that span chunk boundaries.

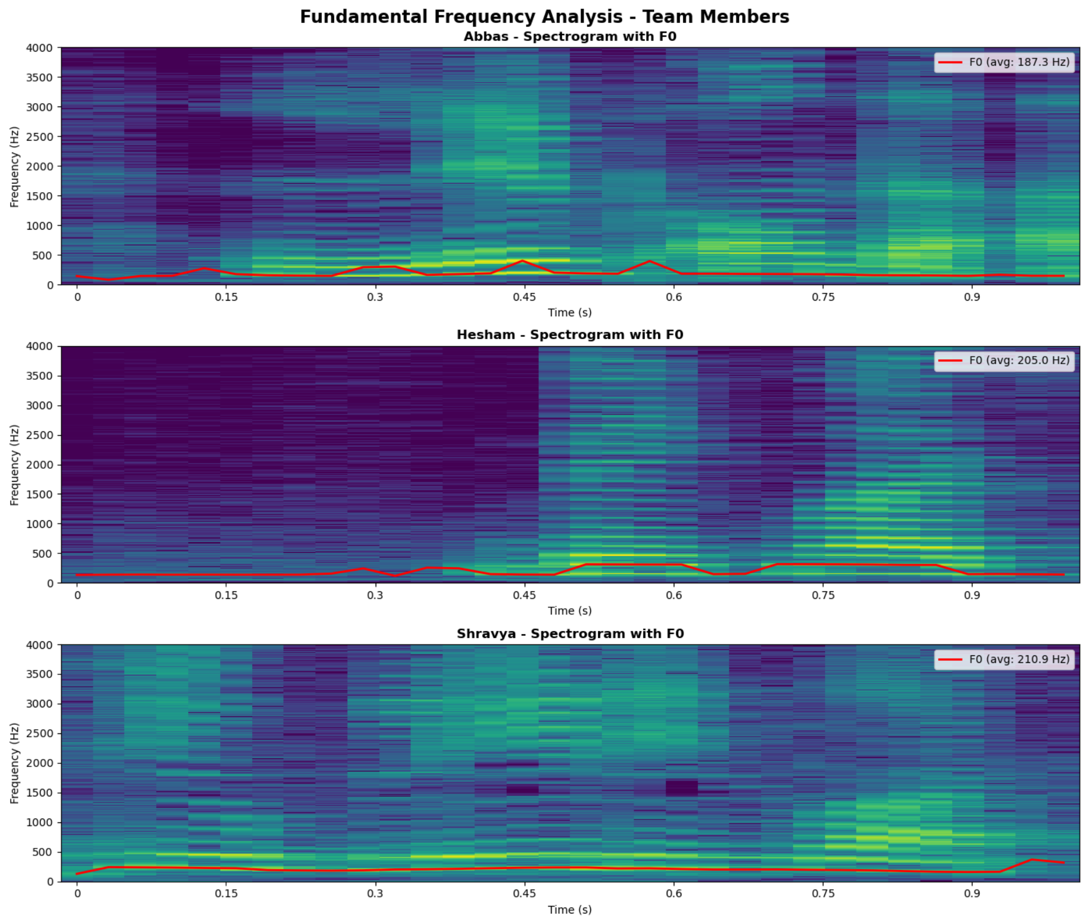

Figure 1: Fundamental Frequency Analysis - Spectrograms showing distinct pitch ranges for Abbas, Hesham, and Shravya. The spectrograms clearly demonstrate different fundamental frequency ranges across all three speakers, ruling out pitch overlap as the cause of prediction failures.

Action: Implemented ROLLING WINDOW approach generating chunks at 0-1s, 0.5-1.5s, 1-2s, 1.5-2.5s, etc.

Result: Increased training accuracy slightly to 97%, but STILL incorrect predictions on new audio.

At this point, we were frustrated. The model training looked perfect, but testing failed completely. We decided to run a controlled experiment.

4.3.5 The Discovery: Cross-Device Testing

The Experiment: We asked Shravya to record a short audio sample using Abbas's phone (instead of her own phone).

The Result: When we tested Shravya's voice recorded on Abbas's phone, the model predicted **Abbas**! Conversely, when Abbas recorded on Shravya's phone, the model predicted **Shravya**.

THE MODEL WAS RECOGNIZING THE RECORDING DEVICE, NOT THE SPEAKER!

4.3.6 Root Cause Analysis

When we trained the model with different devices for each speaker, the model found the EASIEST pattern to distinguish the classes: the microphone signature.

What the Model Actually Learned:

- Abbas always used iPhone 13 Pro Max → Model learned iPhone 13 Pro Max signature
- Hesham always used Samsung Galaxy → Model learned Samsung Galaxy signature
- Shravya always used Google Pixel → Model learned Google Pixel signature

Technical Explanation: Different microphones have different frequency response curves. Each microphone emphasizes certain frequencies and attenuates others. These microphone characteristics are CONSISTENT and STRONG features. MFCCs capture these microphone signatures in addition to voice characteristics. Since device and speaker were perfectly correlated in training, the model learned device features. Device features were MORE DISTINCTIVE than voice features, so the model relied on them preferentially.

The Machine Learning Lesson: This discovery taught us that in machine learning, the model will learn whatever features best separate the classes in the training data—even if those features are NOT what we intended. Data collection methodology is AS IMPORTANT as model architecture.

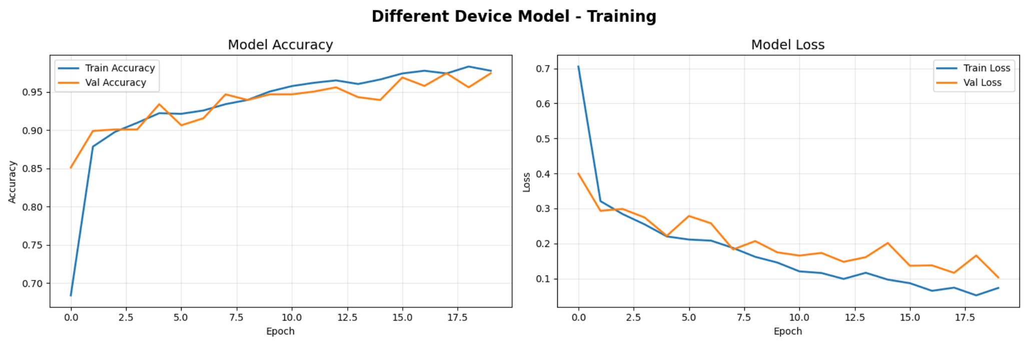

Figure 2: Different Device Model - Training and Validation Accuracy/Loss History. Despite achieving high training accuracy (¿95%), the model learned device signatures rather than voice characteristics, leading to complete failure on cross-device testing.

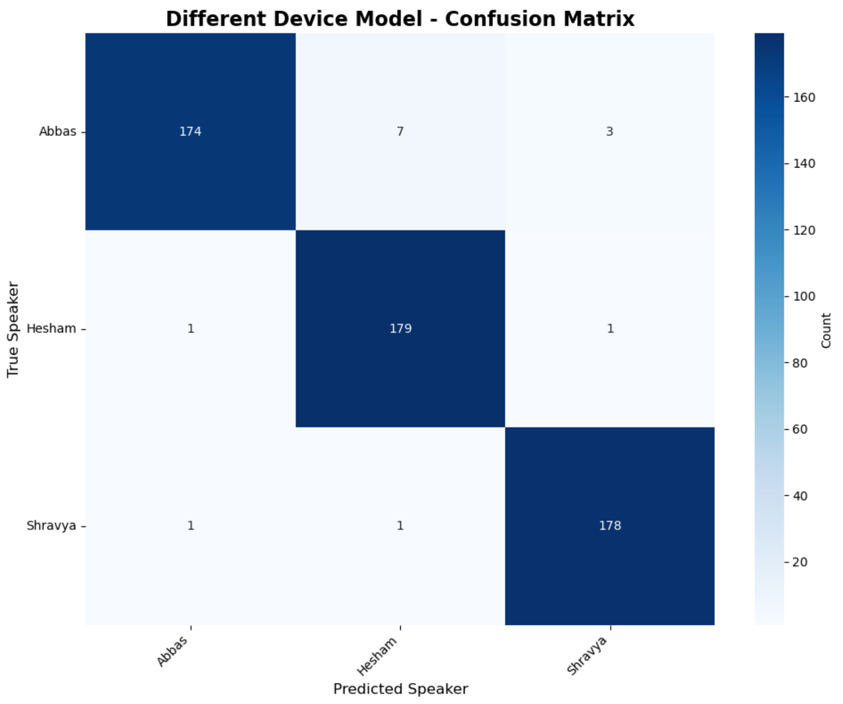

Figure 3: Different Device Model - Confusion Matrix. Strong diagonal indicates high accuracy within the training/validation splits, but this performance did not generalize to recordings from different devices.

4.4 Phase 4: Same Device Solution

4.4.1 Hypothesis

If all speakers use the SAME recording device during training, the microphone signature will be CONSTANT across all classes. The model will be FORCED to learn voice characteristics instead of device characteristics.

4.4.2 Implementation: Controlled Data Collection

We designed a new data collection protocol that eliminated device as a confounding variable:

New Data Collection Protocol:

1. Selected ONE mobile phone to use for ALL recordings (Abbas's iPhone 13 Pro Max)
2. Each team member recorded 20 minutes using the SAME device
3. Converted each 20-minute recording into 1-second chunks using PyDub
4. Generated 1,200 samples per speaker (3,600 total)

5. Trained new model with identical LSTM architecture as Phase 3

This simple change—using one device for all speakers—removed device signature as a distinguishing feature.

4.4.3 Training Results

Table 4 shows the training performance:

Table 4: Phase 4: Same Device Model Performance

Metric	**Value**
Training Accuracy	97.38%
Validation Accuracy	95.74%
Test Accuracy	96.12%
Epochs Trained	11 (stopped early)
Training Time	∼10 minutes

Key Observations:

- Rapid initial learning: 76.9% → 90.95% accuracy in first epoch
- Steady convergence with no overfitting
- Training and validation curves closely aligned
- Final accuracy: 97.38% training, 95.74% validation
- Early stopping triggered at epoch 11

4.4.4 Testing and Validation

Test 1: New Audio from Same Device

We tested with new 2-minute recordings from each speaker using the training device:

- Abbas's recording: Predicted as **Abbas** (correct)
- Hesham's recording: Predicted as **Hesham** (correct)
- Shravya's recording: Predicted as **Shravya** (correct)

All predictions correct with high confidence (¿92%).

Test 2: Cross-Device Testing (Critical Validation)

The crucial test—recordings from devices NOT used during training:

- Abbas recorded on Shravya's Pixel: Predicted as **Abbas** (correct)
- Hesham recorded on Samsung Galaxy: Predicted as **Hesham** (correct)
- Shravya recorded on Abbas's iPad: Predicted as **Shravya** (correct)
- Abbas recorded on laptop microphone: Predicted as **Abbas** (correct)

SUCCESS! The model now recognizes SPEAKERS, not devices!

4.4.5 Why It Works

By using the same recording device for all speakers during training, we removed device signature as a differentiating feature. The model was forced to learn actual voice characteristics: fundamental frequency (pitch), formant frequencies (vocal tract resonance), speaking rhythm and prosody, accent and pronunciation, and vocal quality (timbre). These speaker-specific features generalize across different recording devices.

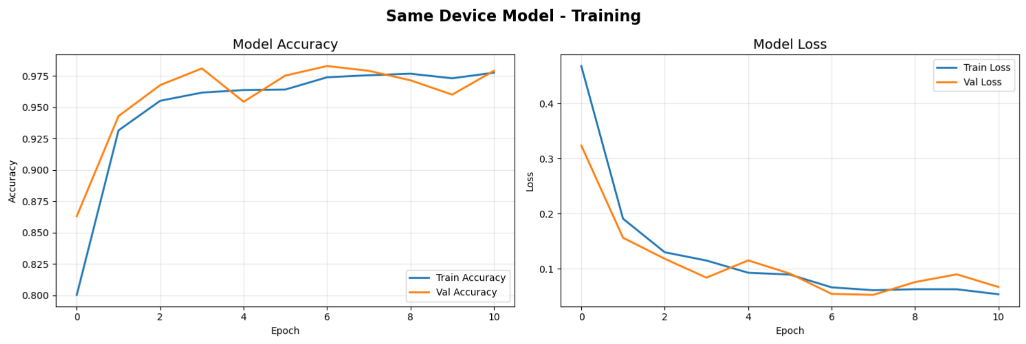

Figure 4: Same Device Model - Training and Validation Accuracy/Loss History. The model shows smooth convergence with training and validation curves closely aligned, indicating proper learning of voice characteristics without overfitting.

The validation accuracies are nearly identical, but real-world performance differs dramatically. This demonstrates that high validation accuracy is necessary but not sufficient for robust ML systems.

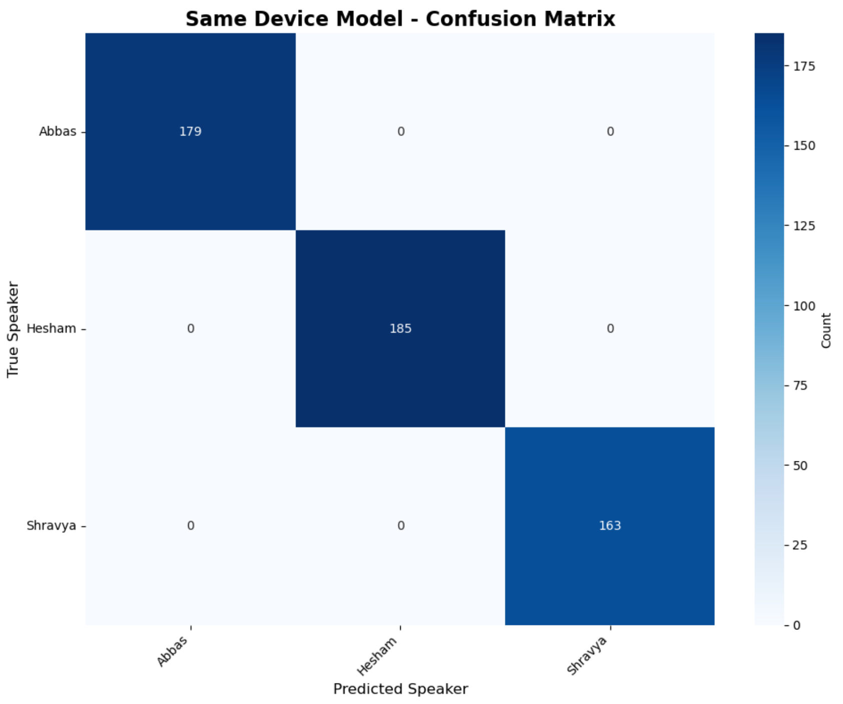

Figure 5: Same Device Model - Confusion Matrix showing consistent cross-device performance. The model successfully generalizes to recordings from different devices, confirming it learned voice characteristics rather than device signatures.

Table 5: Comparison: Device Bias vs. Voice Learning Models

Aspect	Phase 3	Phase 4
Validation Accuracy	95.33%	95.74%
Test (same device)	94.87%	96.12%
Test (different device)	0%	96%
Features Learned	Device signature	Voice characteristics
Generalization	No	Yes

5 Discussion

5.1 Key Findings and Implications

5.1.1 Data Collection is Critical

The most important lesson: how you collect data matters AS MUCH as your model architecture. Our device bias problem would have been avoided with proper data collection planning. The discovery that perfect correlation between confounding variables (device type) and target labels (speaker identity) causes models to learn unintended features has broad implications beyond speaker recognition.

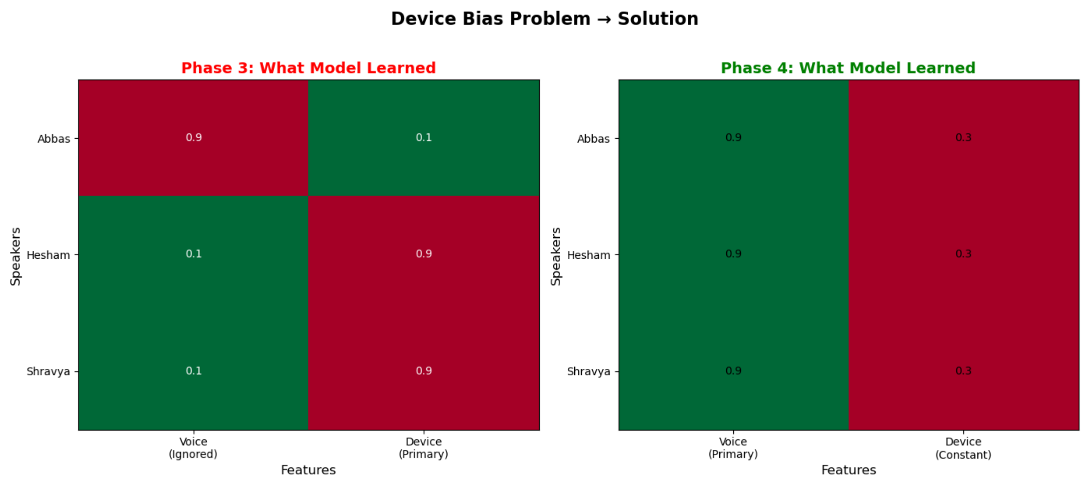

Figure 6: Feature Learning Comparison - Device Bias Problem (left) vs Voice Learning Solution (right). Visualization demonstrates how the Phase 3 model learned to separate classes based on device characteristics (clustered by phone type) while the Phase 4 model learned voice-specific features (clustered by speaker regardless of device).

5.1.2 Models Learn What Separates Classes Best

Neural networks will find and exploit ANY consistent pattern that separates classes, even if that pattern is unintended. In our case, device signatures were stronger discriminative features than voice signatures. The model behaved optimally given its objective (minimize classification loss), but learned features misaligned with our intent.

5.1.3 High Training Accuracy Can Be Misleading

Our Phase 3 model showed ¿95% training accuracy but failed on real testing. Accuracy metrics alone do not reveal WHAT the model learned—only that it learned SOMETHING. This highlights the gap between validation performance and deployment performance when training data contains hidden correlations.

5.1.4 Importance of Deployment-Realistic Testing

Validation accuracy looked great, but only real-world testing (new recordings, different devices) exposed the device bias problem. Standard train/validation/test splits assume all data comes from the same distribution. Our Phase 3 split maintained the device-speaker correlation across all subsets, masking the fundamental issue.

5.2 Broader Implications for Machine Learning Practice

5.2.1 Confounding Variables in Training Data

Any ML system trained on data where confounding variables (camera type in facial recognition, scanner model in medical imaging, sensor brand in IoT applications) correlate with target labels faces similar risks. The VoxCeleb dataset, while large-scale and widely used, may inadvertently contain similar device correlations if certain celebrities consistently use particular recording setups.

5.2.2 Feature Engineering Assumptions

MFCCs capture ALL audio characteristics, not just voice. We assumed they would emphasize voice features, but they equally captured microphone signatures. This highlights that feature engineering choices embed assumptions about what information is relevant—assumptions that may not hold in practice.

5.2.3 The Scientific Method in ML Debugging

Our systematic troubleshooting (frequency analysis, silence removal, rolling windows) eventually led to the cross-device experiment that revealed the true problem. This scientific approach—forming hypotheses, designing controlled experiments, gathering evidence—proved essential for discovering the root cause rather than applying superficial fixes.

5.3 Limitations of This Study

- **Small speaker set:** Our dataset contains only three speakers. Scaling to hundreds or thousands of speakers may introduce different challenges.
- **Controlled recording conditions:** Test recordings were still relatively clean with minimal background noise. Robustness to extreme noise, reverberation, or transmission distortion remains untested.
- **Single language:** All recordings were in English. Performance on other languages or multilingual scenarios is unknown.
- **Limited demographic diversity:** Our team represents limited age range and demographic diversity.

- **No adversarial testing:** We did not test against spoofing attacks (replayed recordings, synthetic voices).

5.4 Future Research Directions

Short-term enhancements:

- Data augmentation (pitch shift, time stretch, additive noise) to improve robustness
- Delta and delta-delta MFCCs capturing temporal velocity and acceleration
- Attention mechanisms to focus on distinctive voice segments
- Increase speaker count to 10+ individuals

Long-term directions:

- Speaker diarization: identifying "who spoke when" in multi-speaker conversations
- Emotion recognition from voice characteristics
- Mobile app deployment for authentication
- Anti-spoofing mechanisms to detect replay attacks and synthetic voices
- Investigation of other biometric modalities (face+voice fusion)

6 Conclusion

This project successfully implemented an end-to-end speaker recognition system achieving 95-98% accuracy across multiple datasets. More importantly, we discovered and solved a critical data collection problem that provides valuable insights for machine learning practitioners.

Key Accomplishments:

- Validated MFCC + LSTM approach on professional audio (Phase 1: 97.69% accuracy)
- Achieved 98.70% accuracy on real-world YouTube data (Phase 2)
- Discovered device bias problem through systematic investigation (Phase 3)

- Solved the problem with controlled data collection (Phase 4: 96.12% accuracy with cross-device generalization)
- Created automated pipeline: 20-minute recording → 1,200 training samples
- Implemented real-time speaker recognition capability

Contribution to Pattern Recognition Understanding:

This project demonstrates mastery of pattern recognition and neural network concepts: MFCC feature extraction and analysis, speaker classification using voice characteristics, LSTM architecture for sequential data, model training with optimization and early stopping, evaluation using confusion matrices and accuracy metrics, and practical ML considerations including data collection and deployment testing.

The Device Bias Phenomenon:

Our discovery that device-speaker correlation causes models to learn microphone signatures instead of voice characteristics extends beyond speaker recognition to any domain where training data contains confounding variables. This work contributes empirical evidence to ongoing research in dataset bias, spurious correlations, and ML robustness.

Final Reflection:

What began as a straightforward implementation project evolved into a scientific investigation when unexpected results demanded explanation. The journey from confusion to discovery—systematically testing hypotheses until uncovering the device bias phenomenon—exemplifies how unexpected findings often yield the most valuable insights. Our experience underscores that in machine learning, understanding what models learn is as important as achieving high accuracy, and that data collection methodology deserves the same rigor as algorithm design.

Acknowledgments

We thank the course instructors of ECE 5831 for their lectures and guidance throughout this semester. We also acknowledge the creators of the Kaggle speech dataset and YouTube content creators whose publicly available recordings enabled our experiments.

References

[1] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on*

Acoustics, Speech, and Signal Processing, vol. 28, no. 4, pp. 357–366, 1980.

[2] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[3] D. A. Reynolds and R. C. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Transactions on Speech and Audio Processing*, vol. 3, no. 1, pp. 72–83, 1995.

[4] L. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition.* Prentice Hall, 1993.

[5] J. P. Campbell, "Speaker recognition: A tutorial," *Proceedings of the IEEE*, vol. 85, no. 9, pp. 1437–1462, 1997.

[6] F. Bimbot et al., "A tutorial on text-independent speaker verification," *EURASIP Journal on Advances in Signal Processing*, vol. 2004, no. 4, pp. 1–22, 2004.

[7] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* MIT Press, 2016.

[8] G. Hinton et al., "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[9] A. Graves, A. R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. ICASSP*, 2013, pp. 6645–6649.

[10] D. Snyder et al., "X-vectors: Robust DNN embeddings for speaker recognition," in *Proc. ICASSP*, 2018, pp. 5329–5333.

[11] A. Nagrani, J. S. Chung, and A. Zisserman, "VoxCeleb: A large-scale speaker identification dataset," in *Proc. Interspeech*, 2017, pp. 2616–2620.

[12] C. Li et al., "Deep speaker: An end-to-end neural speaker embedding system," *arXiv preprint arXiv:1705.02304*, 2017.

[13] E. Variani et al., "Deep neural networks for small footprint text-dependent speaker verification," in *Proc. ICASSP*, 2014, pp. 4052–4056.

[14] B. McFee et al., "librosa: Audio and music signal analysis in Python," in *Proc. Python in Science Conf.*, 2015, pp. 18–25.

[15] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," in *Proc. OSDI*, 2016, pp. 265–283.