Is This Really You? Identifying Key Features for Deepfake Audio Detection Through Deep Models

Samuel Liu, Andrew Kim, Ghali Maata, Sami Nourji
Department of Computer Science, Brown University
{samuel_liu, andrew_kim3, ghali_maata, sami}@brown.edu

I. Introduction

As audio deepfakes become increasingly realistic and less discernible from real, human speech, they threaten the reliability of voice-based authentication systems. While highly accurate detection models do exist, they are computationally expensive and slow, leaving them impractical for accurate, real-time detection systems in settings like phone calls and social media platforms. To design more efficient and lightweight solutions, it is critical to understand which features most effectively distinguish bona fide (real) from spoofed (deepfake) speech. Our aim is to investigate the role of individual MFCC coefficients in deepfake detection. In this paper, we conduct a comparative analysis using the Siamese CNN architecture proposed by Shaaban et al. (2025) [1], which achieved state-of-the-art performance on the ASVspoof 2019 dataset [2]. By combining this architecture with batch-hard triplet loss, this paper aims to learn an embedding space that supports interpretable, feature-level comparisons. In other words, we test how the model's performance changes when each of the 13 MFCC features are ignored to better understand their relative importance for detection.

II. Methodology

A. Extracted features

We used Mel-Frequency Cepstral Coefficients (MFCCs) as input features due to their proven effectiveness in speech-based classification tasks, including deepfake detection. MFCCs approximate how humans perceive sound by emphasizing perceptually relevant frequency bands. They are computed by applying a Discrete Cosine Transform (DCT) to the log-magnitude of the mel-scaled spectrogram, producing a low-dimensional representation of the spectral envelope [3]. We extracted the first 13 coefficients per frame. MFCC 1 captures the overall log energy of the signal, while MFCCs 2 through 13 represent progressively finer spectral details. Lower-order coefficients (e.g., MFCCs 2–5) describe broad frequency content such as vocal tract shape, while higher-order coefficients (e.g., MFCCs 10–13) encode rapid spectral variations that often reflect synthesis artifacts. This balance makes MFCCs particularly useful for detecting fake audio that may match the overall shape of real speech but diverge in high-frequency detail.

[0]Paper Repository: https://github.com/saminourji/DeepfakeAudioDetection

B. Preprocessing

We conduct our experiments using the ASVspoof 2019 LA dataset [2], a standardized training and evaluation dataset for audio deepfake detection. Each audio sample is converted into a 1 x 13 x 400 tensor by extracting the 13 MFCCs. In order to perform convolutions, audio samples of different time lengths were either truncated or padded to have 400 timeframes, which are small, consecutive segments of audio signal. Each tensor is annotated with a speaker ID and a binary label indicating whether the sample is bona fide or spoofed. To reduce computational overhead and enable faster iteration, we apply speaker-balanced subsampling and construct triplets dynamically at the start of each epoch. Each triplet consists of an anchor, a positive sample (same speaker and label), and a negative sample (same speaker, opposite label).

C. Batch-Hard Triplet Mining

During training, we create speaker-balanced batches in which we enforce that all three of the anchor, positive, and negative samples all come from the same speaker label. We then generate sub-batches of the hardest triplets (the exact number of triplets depends on the batch hyperparameters) to update the model's weights. Specifically, using Euclidean distance, we identify the hardest positive (the farthest positive sample from the anchor in the embedding space) and the hardest negative (the closest negative sample to the anchor in the embedding space). This strategy of hard triplet mining is illustrated in Figure 1.

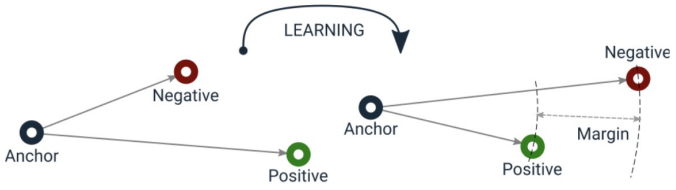

Fig. 1. Illustration of the triplet-loss objective. After the furthest positive and closest negative samples are chosen, the embedding of the anchor is trained to be closer to the positive sample and farther from the negative sample, enforcing class separation in embedding space.

While StacLoss, as introduced in prior work and utilized in Shaaban (2025) [1], is based on contrastive loss over input pairs to separate real and fake audio, our approach instead uses batch-hard triplet loss. This allows us to dynamically select

the hardest positive and negative samples per anchor within each batch, thereby promoting more discriminative feature learning than fixed pairwise loss. This promotes tighter intra-class clustering and greater inter-class separation, consistent with the goals of methods like StacLoss.

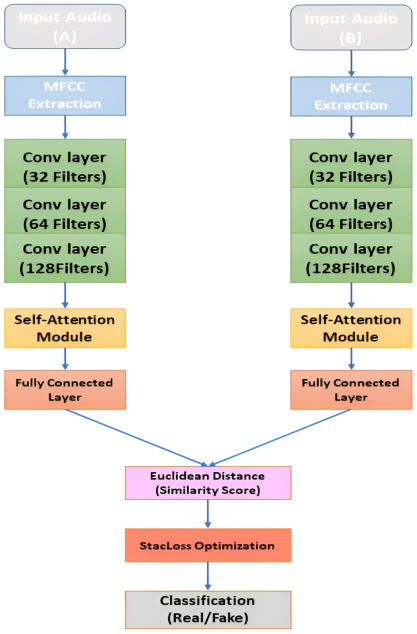

Fig. 2. Overview of the Siamese CNN architecture used in our model. Each branch processes an MFCC spectrogram through three convolutional layers followed by a self-attention module and fully connected layer. Embedding similarity is determined via Euclidean distance and optimized using StacLoss to differentiate between real and fake samples.

D. Batching

We used the following hyperparameters for batching. For each batch:

- **Unique speakers:** 4
- **Minimum spoofed samples:** 2
- **Minimum bonafide samples:** 2
- **Margin:** 1.0 (minimum distance between positive and negative pairs to be able to use for optimizing)

Each speaker thus contributes 8 valid triplets and 4 anchor points, totaling 32 possible triplets and 16 anchor points in the batch. However, our batch-hard triplet loss does not use all of the triplets; instead, for each of the 16 possible anchors, it selects the hardest positive (farthest) and the hardest negative (closest), producing 16 hardest triplets per batch over which the model will optimize.

During hyperparameter tuning, we experimented with different batch composition settings. While this tuning was not guided by a formal search strategy, our choices aimed to control batch size by adjusting the number of hard triplets the model optimized over. We ultimately found that using 4 speakers per batch, each with 2 bonafide and 2 spoofed samples, produced the most effective training behavior, yielding the lowest training loss when using our baseline model (with no features ablated).

E. Model Architecture

Each input to the network is a Mel-spectrogram consisting of 13 frequency coefficients and 400 time frames. The architecture comprises of three convolutional blocks, each applying a 3×3 convolution, followed by batch normalization, ReLU activation, and 2×2 max pooling. To preserve mid-level representations, we introduce a residual connection from the output of the second block into the third. The resulting feature map is passed through a multi-head self-attention mechanism, allowing the model to emphasize the most discriminative regions in the time-frequency space. A final sequence of adaptive average pooling, a fully connected layer, and L_2 normalization yields the embedding. Dropout is used to promote generalization. Two identical branches with shared weights form the Siamese structure. Training is guided by batch-hard triplet loss, encouraging embeddings of the same class to cluster together while maximizing the separation between real and fake audio pairs [1]. An illustration of the model architecture is shown in Figure 2.

F. Feature-ablation protocol

After finalizing the network architecture, we performed a leave-one-feature-out (LOFO) ablation study to assess the relative importance of each MFCC coefficient. We first trained a baseline model using all 13 MFCC features. Then, for each coefficient $k \in \{1, \ldots, 13\}$, we created a modified dataset by removing the k-th feature, resulting in an adjusted input shape of 12×400. Each ablation model was trained under identical settings to the baseline, allowing for a controlled comparison of performance degradation and feature significance.

III. Results

To evaluate the importance of each MFCC coefficient, we conducted a feature ablation study by training 13 models, each omitting one MFCC feature, and compared their performance to the baseline model trained on all 13. The baseline achieved a classification accuracy of 86.60% and an Equal Error Rate (EER) of 13.35%. For each ablation model, we measured the relative drop in accuracy and increase in EER, summarized in Figure 3.

Model	**Δ Acc.** (%)	**Δ EER** (%)
4	−29.0	29.07
12	−26.9	26.89
11	−24.3	24.43
6	−21.7	21.83
2	−17.7	17.66
8	−17.2	17.43
9	−15.6	15.67
13	−14.4	14.59
10	−12.4	12.50
1	−11.6	11.60
7	−9.0	9.00
3	−7.9	8.05
5	−7.4	7.48

Fig. 3. Performance degradation in accuracy and EER when removing individual MFCC coefficients, relative to the baseline model with 86.60% accuracy and 13.35% EER. MFCC 4 and 12 had the most significant impact.

The results show a clear pattern: removing MFCC 4 or MFCC 12 led to the largest performance drops. MFCC 4 caused a 29.0% drop in accuracy and a 29.07% increase in EER, while MFCC 12 caused drops of 26.9% and 26.89%, respectively. These two coefficients thus appear to be critical to model performance. Though there is no universal, fixed mapping from "MFCC 4" or "MFCC 12" to specific frequency bands or concrete perceptual properties, lower MFCCs (e.g. MFCC 4) typically capture mid-frequency spectral information which relates to speech formant structures, while higher coefficients (e.g. MFCC 12) usually reflect fine-grained or high-frequency details. These frequencies are particularly vulnerable to generative attacks in fake audio, making them especially useful for detection.

At the other end, MFCC 5 and MFCC 3 had the smallest impact when removed. The accuracy drop from omitting MFCC 5 was only 7.4%, and the EER increase was 7.48%. MFCC 3 showed similar behavior. These lower mid-order coefficients may contain redundant information already captured by nearby bands, or they may correspond to spectral regions that are less sensitive to deepfake perturbations.

Notably, the drop in accuracy and the increase in EER aligned perfectly in ranking across all features. This consistency supports the validity of both metrics and suggests that MFCC-level feature importance can be reliably estimated through performance degradation (as measured through classification accuracy and EER). The results highlight how specific frequency bands play more important roles in distinguishing bona fide from spoofed speech, and they motivate future work on pruning and compressing models for more efficient deployment. These performance trends are also visualized in Figure 4 and Figure 5, which display the accuracy and EER degradation for each ablation model. Unlike Figure 3, the histograms in Figure 4 and 5 display raw accuracy and EER results for the 13 model variants with different feature ablations, alongside the baseline model.

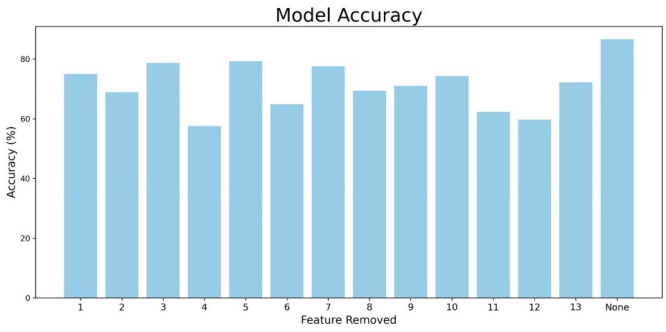

Fig. 4. Histogram of accuracy degradation for each ablation model. Models omitting MFCC 4 and 12 show the largest drop, while those omitting MFCC 3 and 5 show minimal impact.

IV. Challenges

A key implementation challenge arose during the integration of batch-hard triplet loss from Shaaban (2025) [1].

Fig. 5. Histogram of Equal Error Rate (EER) for each ablation model. Models omitting MFCC 4 and 12 show the largest drop, while those omitting MFCC 3 and 5 show minimal impact.

While the concept of hard mining was clearly described, the paper omitted critical implementation details—most notably, whether triplets should be mined globally across the entire epoch or locally within each batch. The authors appeared to favor global mining, which requires computing a full pairwise distance matrix and leads to a quadratic cost in dataset size.

Moreover, computing every valid triplet across the full dataset scales cubically with the dataset size, roughly $O(N^3)$, because for each anchor we must consider all valid positives and all possible negatives. This strategy, while theoretically appealing, would have been computationally infeasible for our use case, especially given that we needed to retrain the model 14 times for feature ablation (Shaaban (2025) [1] reports 20+ hours to train a single model).

To address this, we opted for a batch-level mining strategy. As described in our preprocessing pipeline, we selected hard positives and negatives dynamically within each speaker-balanced batch. Indeed, computing all valid hardest triplets within batches is far more computationally efficient (reducing runtime as well as memory usage) while still providing meaningful gradients for training.

To stabilize training under this lightweight regime, we increased the number of epochs from 100 to 500 and adjusted our batching hyperparameters. This adaptation proved effective, allowing us to train each model variation efficiently while maintaining competitive performance.

V. Reflection

A. How do you feel your project ultimately turned out? How did you do relative to your base/target/stretch goals?

We consider the project a success. We were able to train our model effectively and obtain interpretable results that could inform practical applications. Although our original plan was just to reimplement an existing paper, we pivoted to contribute original insights by evaluating the relative importance of MFCC features. While our base, target, and stretch goals shifted, our final outcome fits what we would now consider a target-level goal. We were able to determine a relative ranking of each of 13 MFCC coefficients/features, though

improvements could definitely be made to reach a stretch goal as detailed later.

B. Did your model work out the way you expected it to?

Since our goal was to compare models under different feature ablations, we trained multiple models and compared their results under standardized conditions. To meet these requirements, we chose a lightweight training setup to reduce training time, which we anticipated would come at the cost of total performance in accuracy during the evaluation stage. Still, our baseline model reached a solid 86.60% accuracy and 13.35% equal error rate. And as expected, the models with one MFCC coefficient removed each performed worse than the baseline. However, we were surprised by the wide variance in performance drops across features, suggesting that certain MFCC bands are far more informative for deepfake detection than others. In particular, as mentioned in the results section, MFCC 4 and MFCC 12 seemed to be very important for deepfake detection.

C. How did your approach change over time? What kind of pivots did you make, if any? Would you have done differently if you could do your project over again?

We made a significant pivot in our project. Initially, our aim was to develop a real-time detection system that outputs a continuous probability of an audio stream being a deepfake, updating over time (based off the same Siamese CNN model). However, we realized that this required complex sequential modeling and high computational overhead, which were not feasible within our timeline. Instead, we shifted focus to a more tractable but still impactful question: identifying which MFCC features are most informative for audio deepfake detection. This allowed us to contribute new insights while building on state-of-the-art methods like Shaaban et al.'s Siamese CNN [1]. If we were to do the project again, we might have made this pivot earlier. That would have given us more time to apply our findings by experimenting with lighter models using only the top-ranked MFCCs for real-time deployment, which was the original goal of this project.

D. What do you think you can further improve on if you had more time?

With more time and resources, we would have extended our ablation study by testing combinations of features, not just one-by-one removals. This would have allowed us to identify patterns between MFCCs and better isolate redundant features. Ideally, we would then distill this knowledge into a new, highly compressed model using only the most essential coefficients. For instance, a feasible next step would be to continue feature ablation with different combinations of MFCC features until a certain accuracy threshold was met (e.g. if we reach less than 60% classification accuracy, then consider the ablated features as "very important"). Given more compute, we would have fully trained this lightweight variant and benchmarked it against our baseline, evaluating trade-offs in speed, memory, and accuracy. This would bring us closer to real-time deployment, which remains an important long-term goal.

E. What are your biggest takeaways from this project/what did you learn?

Our biggest takeaway is that deepfake detection is both challenging and highly relevant. It sits at the intersection of technical complexity and real-world impact, and we were excited to contribute to a problem with clear industry applications. Through this project, we deepened our understanding of audio preprocessing, model training, and experiment design. From a technical standpoint, version control via Git and modular repo design were crucial to help the group stay organized and work on the project simultaneously. We also learned how to scope a comparative study that yields interpretable results under tight constraints, which is a skill that will carry over to future machine learning work.

REFERENCES

[1] O. A. Shaaban and R. Yildirim, "Audio deepfake detection using deep learning," *Engineering Reports*, vol. 7, no. 3, p. e70087, 2025. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/eng2.70087

[2] X. Wang, J. Yamagishi, M. Todisco, H. Delgado, A. Nautsch, N. Evans, M. Sahidullah, V. Vestman, T. Kinnunen, K. A. Lee, L. Juvela, P. Alku, Y.-H. Peng, H.-T. Hwang, Y. Tsao, H.-M. Wang, S. L. Maguer, M. Becker, F. Henderson, R. Clark, Y. Zhang, Q. Wang, Y. Jia, K. Onuma, K. Mushika, T. Kaneda, Y. Jiang, L.-J. Liu, Y.-C. Wu, W.-C. Huang, T. Toda, K. Tanaka, H. Kameoka, I. Steiner, D. Matrouf, J.-F. Bonastre, A. Govender, S. Ronanki, J.-X. Zhang, and Z.-H. Ling, "Asvspoof 2019: A large-scale public database of synthesized, converted and replayed speech," 2020. [Online]. Available: https://arxiv.org/abs/1911.01601

[3] 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.