# Audio Deepfake Detection *aliniere*

# 1. State of the Art

In recent years, the rise of voice-based authentication systems has sparked a parallel interest in audio spoofing attacks, where synthetic or replayed audio can impersonate legitimate users. Detecting such spoofing attempts has become critical. Traditional methods relied on feature engineering using spectral or phase-based features; however, deep learning models, particularly Convolutional Neural Networks (CNNs), have proven to outperform classical methods by automatically extracting hierarchical features from audio spectrograms [1][2].

The ASVSpoof2019 challenge [3] provided a standard benchmark dataset that stimulated the development of robust models for logical access (LA) attacks. Modern solutions typically involve processing the audio into time-frequency representations, such as mel-spectrograms, and training CNN-based classifiers. State-of-the-art systems have incorporated various strategies, including data augmentation, ensemble learning, and calibration techniques to further improve reliability [4][5].

The proposed CNN model is tailored for audio spoofing detection using mel-spectrograms as input. It is intentionally lightweight to ensure fast inference and high accuracy, making it suitable for deployment on resource-constrained devices. Earlier work has also compared a wide range of handcrafted spectral and cepstral features for spoofing detection, highlighting their relative strengths in traditional systems [6].

**Network Structure** The model takes as input a preprocessed mel-spectrogram of fixed size (128 x 109 x 1), where 128 represents mel bands and 109 the number of time steps.

The first convolutional block applies a Conv2D layer with 32 filters and a (3x3) kernel, followed by a ReLU activation to extract local features like harmonics and time-frequency patterns. A MaxPooling2D layer (2x2) reduces spatial dimensions and computational load.

The second convolutional block includes another Conv2D layer with 64 filters and ReLU activation, followed by another (2x2) max pooling layer to capture more abstract patterns.

Feature maps are then flattened into a 1D vector. A fully connected Dense layer with 128 neurons and ReLU activation processes the extracted features. A Dropout layer with a 50% rate is applied to prevent overfitting and enhance generalization.

Finally, the output layer consists of 2 neurons (for spoof and bonafide classes) with a Softmax activation to produce normalized probabilities for classification.

**Design Rationale** Using small (3x3) filters helps capture fine-grained local structures, crucial for distinguishing subtle differences in spoofed vs. genuine audio. Pooling layers provide translational invariance and reduce dimensionality. Dropout improves robustness against overfitting, especially under adversarial conditions. The Softmax output supports binary classification and enables threshold-based decision making.

This architecture aligns with recent research emphasizing lightweight CNNs for spoofing detection [2][5], offering a strong balance between efficiency and detection performance. Cross-database evaluations have further shown the limitations of model generalization when systems are trained and tested on different datasets [7].

# 3.Evaluation Metrics

### 3.1 ROC Curve and AUC

### 

### Fig. 1

### The ROC curve in Figure 1 shows the model's classification performance across different thresholds. AUC reaches 0.90, indicating strong separability between spoofed and bonafide samples. The sharp rise toward the top-left suggests high accuracy, while the stepped shape may result from limited threshold sampling during evaluation.

### 3.2 Precision-Recall Curve

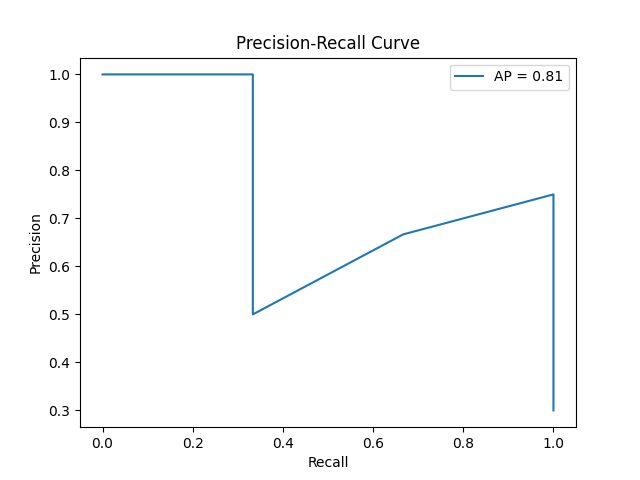


Fig. 2

In Figure 2, given the class imbalance often present in real-world spoofing datasets, the Precision-Recall (PR) curve provides a more informative view than ROC. The average precision (AP) score obtained is **0.87**, showing high precision even at varying recall levels.

### 3.3 Calibration Curve

Fig. 3

In Figure 3 calibration ensures predicted probabilities reflect true outcome likelihoods. In the plot, the blue line represents the model's predictions, while the dashed diagonal shows perfect calibration.

The curve indicates that the model tends to be overconfident, especially in the mid-range. While it separates classes well, probability estimates could be improved using methods like **Platt Scaling**.

### 3.4 Equal Error Rate (EER)

The Equal Error Rate (EER) calculated from the ROC curve is **approximately 0.13**, which is a competitive result compared to similar CNN-based systems in literature [1][3].

## Implementation Details

The dataset contains approximately 20,000 audio files in English. Each bonafide utterance has a corresponding spoofed version, making it a parallel dataset. This ensures that the spoof detection focuses on voice characteristics rather than linguistic content.

The system implementation was partially based on the publicly available repository at GitHub [10], which provided a foundation for preprocessing, model training, and evaluation components.

The complete system for training, evaluating, and predicting has been implemented using **TensorFlow** and **Keras**, following best practices for machine learning pipelines. The code is organized into two main Python scripts:

main.py — Training and evaluation

predict.py — Single-file inference and visualization

The implementation workflow consists of the following key stages:

### 4.1 Preprocessing

All audio files are loaded and resampled to a uniform **16 kHz** sample rate using librosa. Each audio clip is truncated or padded to exactly **5 seconds**. Afterward, a **mel-spectrogram** with **128 mel bands** and **109 time frames** is computed for each sample.  
The power spectrogram is converted to decibels (dB) for better feature representation, and the resulting spectrogram is normalized and reshaped to fit the CNN input format (128 × 109 × 1).

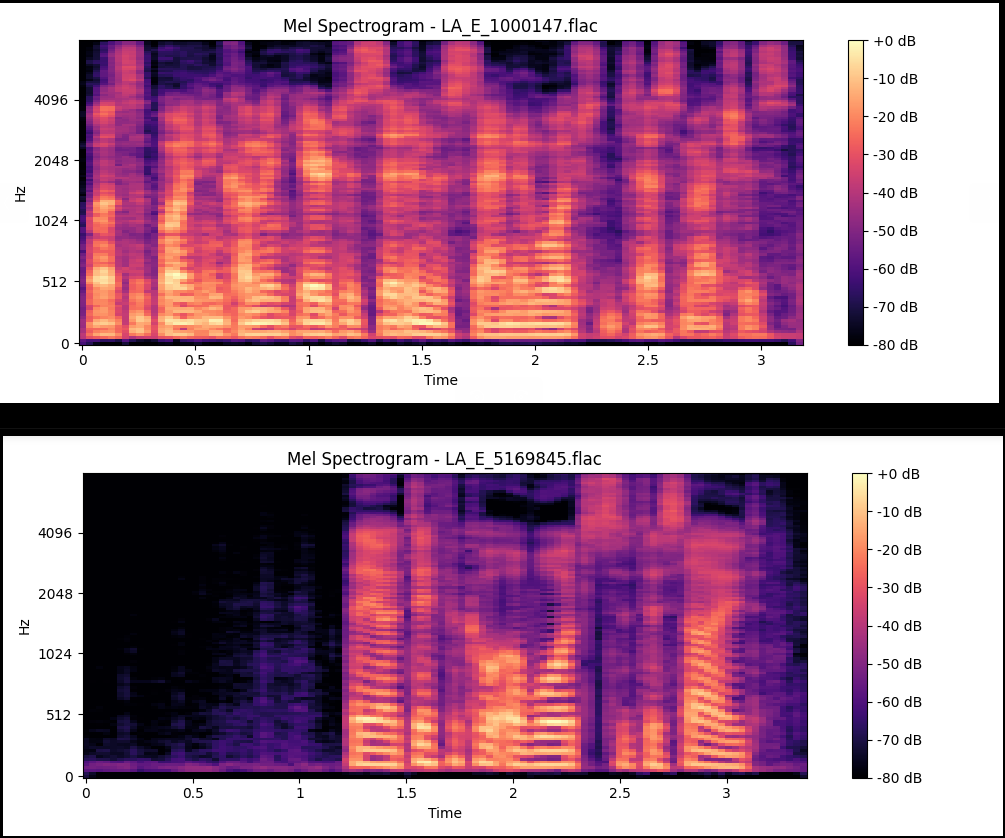


Fig. 4

**Mel spectrograms** visualize the frequency content of an audio signal over time using a perceptual Mel scale. In Figure 4, the top spectrogram (LA\_E\_1000147.flac) represents a **spoof** sample, while the bottom one (LA\_E\_5169845.flac) is a **bonafide** (genuine) recording. The spoof signal shows an abrupt start and more uniform, less detailed spectral patterns. In contrast, the bonafide spectrogram has smoother transitions and richer frequency variations, reflecting more natural speech dynamics.

This preprocessing ensures consistency across the dataset and facilitates better learning by the CNN model.

### 4.2 Training

The available dataset is split into **80% training** and **20% validation** subsets.  
The model is trained from scratch with the following setup:

**Optimizer:** Adam

**Loss function:** Categorical Crossentropy

**Metrics:** Accuracy

**Batch size:** 32

**Epochs:** 10

**Dropout rate:** 0.5 to mitigate overfitting

Adjusting the dropout rate impacts the model's ability to generalize. Reducing it to **0.3** means fewer neurons are dropped during training, allowing the model to learn more detailed patterns, but increasing the risk of overfitting. Increasing it to **0.7** enforces stronger regularization by dropping more neurons, which can help prevent overfitting but may also lead to underfitting and slower convergence. The commonly used rate of **0.5** typically balances these effects, offering both sufficient learning capacity and regularization.

Training history (loss and validation loss) is recorded and plotted, offering insights into the model’s convergence behavior. If a pre-trained model (audio\_classifier.h5) is detected, it is loaded directly to save time. Otherwise, the model is trained and saved.

### 4.3 Evaluation

After training, the model undergoes extensive evaluation using multiple metrics:

**ROC Curve and AUC Calculation:**  
To assess the model’s ability to discriminate between classes across thresholds.

**Precision-Recall (PR) Curve:**  
To measure precision and recall at various thresholds, particularly important in unbalanced datasets.

**Calibration Curve:**  
To verify if the predicted probabilities are well-calibrated.

**Confusion Matrix:**  
To visualize classification errors between spoof and bonafide samples.

**Equal Error Rate (EER):**  
Calculated to benchmark against standard spoofing detection systems.

All plots are automatically saved under a /plots directory for later analysis.

### 4.4 Prediction Script

A separate script, predict.py, provides a simple interface for evaluating individual .flac files.  
The prediction workflow is as follows:

* Load the chosen file.
* Preprocess it identically to the training stage.
* Predict the class label (spoof or bonafide) using the trained CNN model.
* Output the class label with associated confidence score.
* Generate and save the mel-spectrogram visualization for inspection.

This separation of training and inference stages adheres to software engineering best practices, ensuring maintainability and reproducibility.

The architecture illustrated in Figure 5 presents a lightweight Convolutional Neural Network (CNN) developed for audio spoofing detection. It consists of two convolutional blocks incorporating ReLU activations and max pooling layers for hierarchical feature extraction and spatial reduction. These are followed by a fully connected layer with dropout regularization to enhance generalization, and a final softmax layer that outputs normalized class probabilities for bona fide and spoofed audio inputs.

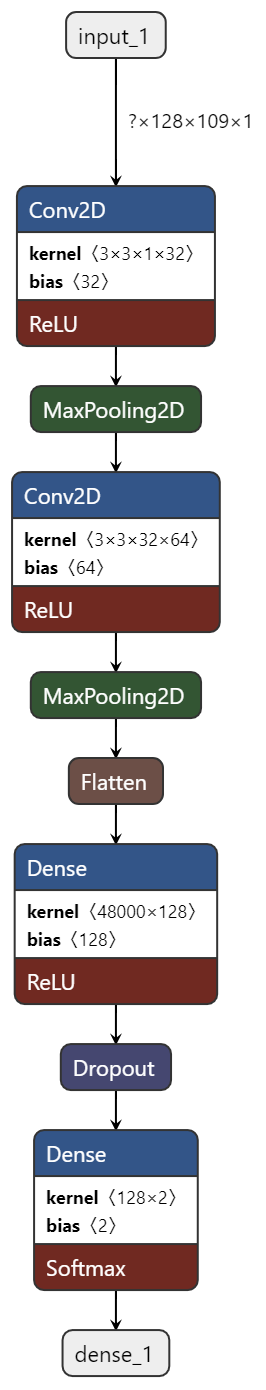


Fig. 5

## 5. Conclusion

This study demonstrates that a relatively simple and lightweight Convolutional Neural Network (CNN) architecture can deliver strong performance in the task of audio spoofing detection. Despite its compact design, the model achieves competitive results across key evaluation metrics:

* Area Under the ROC Curve (AUC): 0.90
* Average Precision (AP): 0.87
* Equal Error Rate (EER): 0.13

These outcomes highlight the model’s ability to effectively distinguish between bona fide and spoofed audio samples, confirming the suitability of mel-spectrograms combined with a two-block CNN structure for this task.

While the results are promising, there is still room for improvement. Techniques such as ensemble learning—where multiple models are combined—may further enhance both robustness and accuracy. Likewise, applying advanced probability calibration methods (e.g., Platt scaling or isotonic regression) could improve the reliability of the model’s outputs, which is particularly important in security-sensitive applications.

Future work may involve exploring more sophisticated network architectures, such as Residual Networks (ResNets) [4], known for their superior feature extraction capabilities. Additionally, incorporating data augmentation techniques—such as noise injection, pitch shifting, or time stretching—could improve the model’s generalization to various spoofing scenarios.

Finally, evaluating the system on larger and more diverse datasets beyond ASVSpoof2019 would offer a more comprehensive understanding of its robustness and real-world applicability. End-to-end architectures such as RawNet2 have also shown promising results in spoofing detection by directly learning from raw waveform inputs [8].

# References

[1] J. Todisco, H. Delgado and N. Evans, "Constant Q Cepstral Coefficients: A Spoofing Countermeasure for Automatic Speaker Verification," IEEE Transactions on Information Forensics and Security, vol. 12, no. 4, pp. 903-917, April 2017. doi: 10.1109/TIFS.2016.2625257

[2] A. Nautsch, X. Wang, M. Todisco, et al., "Spoofing Detection and Countermeasures for Automatic Speaker Verification: A Survey," IEEE Signal Processing Magazine, vol. 39, no. 4, pp. 82-97, July 2022. doi: 10.1109/MSP.2022.3168905

[3] X. Wang, H. Delgado, A. Nautsch, et al., "ASVspoof 2019: A Large-Scale Public Database and Challenge for Synthetic Speech Detection," Proc. Interspeech 2019, pp. 1488-1492, 2019. doi: 10.21437/Interspeech.2019-1850

[4] Y. Jiang, X. Chen, J. Liu, et al., "An Improved ResNet-Based Approach for Spoofing Detection," IEEE Access, vol. 8, pp. 151810-151820, 2020. doi: 10.1109/ACCESS.2020.3016971

[5] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-Vectors: Robust DNN Embeddings for Speaker Recognition," Proc. ICASSP 2018, pp. 5329-5333, 2018. doi: 10.1109/ICASSP.2018.8461375

[6] M. Sahidullah, T. Kinnunen and C. Hanilçi, "A Comparison of Features for Synthetic Speech Detection," Proc. Interspeech 2015, pp. 2087-2091, 2015. doi: 10.21437/Interspeech.2015-472

[7] P. Korshunov and S. Marcel, "Cross-Database Evaluation of Deep Neural Networks for Spoofing Detection," Proc. Interspeech 2017, pp. 84-88, 2017. doi: 10.21437/Interspeech.2017-1392

[8] H. Tak, J. Patino, N. Evans and J. Yamagishi, "End-to-End Anti-Spoofing with RawNet2," Proc. ICASSP 2021, pp. 6369-6373, 2021. doi: 10.1109/ICASSP39728.2021.9413682

[9] H. Delgado, M. Todisco, and N. Evans, "ASVspoof 2021: Automatic speaker verification spoofing and countermeasures challenge evaluation plan," arXiv preprint, arXiv:2109.00535, 2021.

[10] sksmta, “audio-deepfake-detection,” GitHub. [Online]. Available: https://github.com/sksmta/audio-deepfake-detection [Accessed: 2-May-2025].