Baby Saral G, 2026, 14:2
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Deepfake Voice Detection Using CNN–BiLSTM Architecture

Baby Saral G, Vibin Dev Anand, Hariharan A, Nehaansh Ladiwala
Dept. of Comp. Intelligence SRM Inst. of Sci. & Tech. Chengalpattu, India

Abstract: Voice synthesis has come a long way. In just a few years, AI-generated speech has gone from obviously robotic to genuinely convincing—convincing enough that distinguishing a real recording from a synthetic one is no longer something a casual listener can reliably do. That shift creates a real problem for voice authentication systems, digital media verification, and any context where the authenticity of an audio recording actually matters. In this work, we explored whether a hybrid CNN–BiLSTM architecture, operating on Mel spectrogram representations, could reliably separate genuine human speech from AI-generated audio. The idea was to let convolutional layers pick up on spectral irregularities that synthesis systems tend to leave behind, while bidirectional LSTM layers model how those patterns evolve across time—something a frame-by-frame analysis would miss entirely. We trained and evaluated the model using the Fake-or-Real (FoR) dataset, which contains around 17,870 labelled clips split evenly between authentic recordings and outputs from various neural TTS systems. On the held-out test partition, we observed an overall accuracy of 99.02%, with precision, recall, and F1-score each sitting at 0.98, and an AUC of 0.9998. Honestly, the AUC was higher than we anticipated going in. To check whether the recurrent component was actually helping, we also ran a CNN-only version under identical conditions—it dropped to 96.4% accuracy, which suggests the temporal modelling is doing something the convolutional layers alone cannot. We describe the full pipeline here, from raw audio input through to REST API deployment, and we try to be upfront about where this approach currently falls short.

Keywords: Audio Forensics, Audio Preprocessing, CNN–BiLSTM, Deepfake Voice Detection, Mel Spectrogram, Speech Classifica-tion, Synthetic Speech.

I. INTRODUCTION

A voice carries a lot of weight. We use it to prove identity, sign off on financial decisions, and give testimony that courts take seriously. The assumption baked into all of that is straightforward: a voice recording represents the person who produced it. For most of history, that assumption held. It is holding less and less now.

Neural speech synthesis has improved dramatically over a short period. Systems built on generative adversarial networks, flow-based architectures, and more recently diffusion models can clone a speaker's voice from just a few seconds of reference audio, producing output that is—in many cases—indistinguishable from the original even to trained ears (Goodfellow et al., 2014; van den Oord et al., 2016). The consequences have already moved from theoretical to documented: there are recorded cases of phone fraud using cloned voices, synthetic audio of politicians being circulated as real, and at least a handful of incidents where voice-based banking authentication was bypassed using AI-generated speech. This is not a future threat. It is a current one.

Building automated detectors for synthetic speech has therefore become a genuinely important problem. Deep learning has proven effective across a wide range of tasks involving perceptual signals (LeCun et al., 2015), and the intuition that it might generalise well to audio forensics is borne out in the literature. The harder

challenge is that most detection systems are implicitly tuned to the synthesis methods present in their training data. A new vocoder architecture can break a detector that worked well on the previous generation. That fragility is a recurring theme, and it is worth keeping in mind when reading any benchmarked detection result—including ours.

In this work, we approached the problem using log-Mel spectrograms as input to a CNN–BiLSTM classifier. Convolutional layers handle spectral patterns; the BiLSTM handles how those patterns move through time. We evaluated on the FoR benchmark and found that the combination works well, though the caveats about generalisation still apply. The rest of this paper is laid out as follows: Section 2 covers related work; Section 3 describes the dataset; Section 4 goes through preprocessing; Section 5 explains the architecture and training; Section 6 covers deployment; Section 7 presents and discusses results; and Section 8 concludes.

II. RELATED WORK

Classical Feature-Based Approaches

Research into detecting synthetic speech did not really coalesce as a distinct subfield until the ASVspoof challenge series began organising it in 2015 (Wu et al., 2015). Before then, most relevant methods were borrowed from speaker verification. ASVspoof brought structure: shared datasets, common evaluation protocols, and a community of researchers all working on the same problem. Early submissions relied heavily on handcrafted features—MFCCs, constant-Q cepstral coefficients, sub-band spectral flux—fed into standard classifiers like Gaussian Mixture Models and SVMs.

Those systems worked reasonably well within the boundaries of each challenge. The issue was portability. A detector trained to recognise the stitching artefacts of unit-selection concatenative synthesis has almost nothing useful to say about a WaveNet-style vocoder, which operates on completely different principles and produces a completely different acoustic signature (Korshunov and Marcel, 2019). The features were too narrowly tied to particular synthesis mechanisms. Each new generation of synthesis technology essentially required a new detector.

Deep Learning Approaches

Things began to shift noticeably when researchers started applying convolutional networks directly to spectrogram represen-tations rather than to pre-extracted feature vectors. By around 2018–2019, CNN-based systems were regularly outperforming the best classical entries on ASVspoof 2019 (Alzantot et al., 2019; Todisco et al., 2019). The advantage is fairly intuitive—instead of computing a fixed set of features and hoping they are the right ones, a CNN learns from the data what patterns are actually discriminative. That tends to produce more general detectors, at least within the distribution of the training set.

Recurrent networks added a complementary perspective. LSTMs, in particular, seemed well suited to capturing the temporal irregularities that synthetic speech sometimes exhibits—slight prosodic unnaturalness, inconsistent phoneme transitions, or frame-level artefacts that only become apparent in sequence (Hochreiter and Schmidhuber, 1997). Running the LSTM bidirectionally means the model sees both past and future context at each time step, which is arguably more appropriate for speech signals where local naturalness depends on what comes before and after. Self-attention mechanisms have also been shown to model long-range sequential dependencies effectively (Vaswani et al., 2017), and a number of recent papers have explored hybrid architectures combining convolutional, recurrent, and attention-based components. Whether those additions consistently improve detection remains an open question, but the general direction seems promising.

Spectrogram-Based Classification

Among the various input representations that have been tried—raw waveforms, MFCCs, various cepstral coefficients—Mel spectrograms have become the most

commonly used for CNN-based deepfake detection. Part of this is practical: spectrograms produce a two-dimensional representation that slots naturally into convolutional architectures. Part of it is perceptual: the Mel scale approximates the frequency resolution of the human auditory system, so regions of the spectrum where speech characteristics are most prominent get proportionally more detail.

Empirically, comparisons tend to favour Mel spectrograms over MFCC inputs for this task, likely because spectrograms retain the full spectral picture rather than compressing it down to a coefficient vector (Tak et al., 2022). Whether that extra information is always necessary, or whether a more compact representation would suffice for simpler detection tasks, is probably context-dependent.

Limitations of Existing Methods

Reading across this literature, a few patterns keep recurring. Overfitting to specific synthesis methods is nearly universal—most detectors degrade when tested on TTS systems they were not trained on, sometimes substantially. Real-world audio conditions are largely underexplored: telephone compression, environmental noise, and codec distortion all appear infrequently in published evaluations, which tend to use clean studio recordings. And the vast majority of systems are offline classifiers—they require a complete audio file and cannot operate on a live stream. We addressed the streaming limitation through our deployment architecture, but we did not solve the first two issues, and we will revisit this honestly in the limitations section.

III. DATASET

For training and evaluation, we used the Fake-or-Real (FoR) dataset. It has become a reasonably standard benchmark in this space, which is partly why we chose it—comparability matters. The dataset contains approximately 17,870 audio clips, balanced between genuine human recordings and synthetic outputs from several neural TTS pipelines, including WaveNet-based and Tacotron-based systems (van den Oord et al., 2016). Each clip is around two seconds long. That fixed duration is convenient because it means every sample maps to the same spectrogram dimensions without padding or truncation, which simplifies the data pipeline considerably.

The recordings cover a reasonable spread of speakers, though it is worth noting that all of the audio is fairly clean—there is no telephone-quality material, no background noise, and no heavy compression artefacts. That is a limitation of the dataset that carries through to any model trained on it. We split the data 80/10/10 for training, validation, and testing respectively, keeping class balance consistent across all three splits. Running imbalanced splits would risk inflating accuracy metrics, so this was a deliberate choice.

IV. AUDIO PREPROCESSING

Normalisation and Resampling

The FoR clips arrive at inconsistent sample rates, so the first step was resampling everything to 22,050 Hz. After that, we applied peak amplitude normalisation to each waveform. It is a simple step, but skipping it would allow loudness variation to bleed into the spectrogram features—and a model that picks up on recording level as a cue is learning something spurious. This likely matters more for cross-corpus generalisation than for within-dataset performance, but it is good practice regardless.

Mel Spectrogram Extraction

Feature extraction used the Librosa library (McFee et al., 2015). We computed short-time Fourier transforms with a 2,048-sample Hann window and a hop length of 512 samples, then projected the power spectrogram onto 128 Mel-scale frequency bands covering 0–11,025 Hz. Log (dB) scaling was applied to compress dynamic range. For a two-second clip at 22,050 Hz, this reliably produces a 128 × 87 × 1 tensor—128 frequency bins, 87 time frames, single channel—which feeds directly into the first convolutional layer.

One decision we made early on was not to apply any data augmentation during preprocessing. The concern was that augmentation techniques—pitch shifting, time stretching, adding noise—could introduce artefacts of their own, and if those artefacts correlate with either class, the model might learn to exploit them rather than the actual acoustic properties we care about. In a cleaner experimental setup, this is probably the right call, even if it limits the training set size.

V. METHODOLOGY

Model Architecture

The core idea behind the architecture (Figure 1) is that catching synthetic speech requires attending to two different things simultaneously: what the spectrum looks like at any given instant, and how it changes across time. Convolutional layers are well suited to the first. The BiLSTM handles the second.

The convolutional portion consists of three blocks, each following the same pattern: a 3 × 3 2-D convolution, then batch normalisation, ReLU activation, 2 × 2 max-pooling, and dropout at rate 0.25. The filter counts step up through 32, 64, and 128 across the three blocks—a fairly standard progressive widening that lets the network build from low-level spectral edges toward more abstract patterns. After the third block, a reshape operation collapses the frequency and channel dimensions into a sequence of 128-dimensional vectors indexed along the time axis. This is the bridge between the convolutional and recurrent parts: the BiLSTM sees a temporal sequence of CNN-extracted feature vectors rather than a static image.

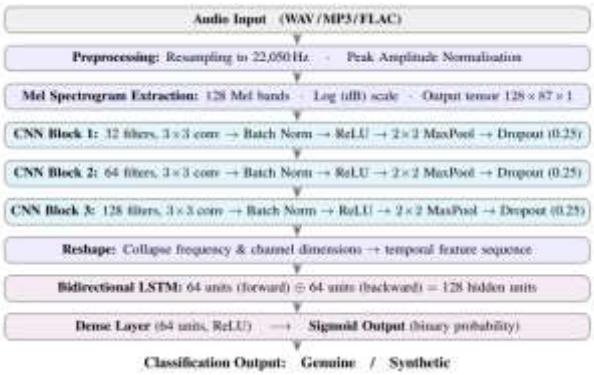
The BiLSTM runs 64 units in each direction, for 128 units total once the forward and backward outputs are concatenated. That concatenation flows into a dense layer of 64 units with ReLU activation, and then into a single sigmoid output node. Total parameter count is approximately 1.2 million. We kept the model relatively small on purpose—partly to avoid overfitting on a dataset of this size, partly because we wanted something that could realistically be deployed without specialised hardware.

Training Configuration

We trained using binary cross-entropy loss with the Adam optimiser (Kingma and Ba, 2015), starting at a learning rate of 1 × 10−3. A learning-rate scheduler monitored validation loss and reduced it by a factor of 0.5 if no improvement was seen for five consecutive epochs—this helped smooth out the late-stage oscillations that sometimes show up with Adam. Training ran for a maximum of 50 epochs with a batch size of 32, but early stopping with a patience of 10 epochs on validation loss meant it typically terminated sooner. The best-validation checkpoint was used for all reported results. Everything ran on Google Colab with GPU acceleration; each epoch took roughly 45 seconds.

Evaluation Metrics

We report accuracy, precision, recall, F1-score, and AUC. Of these, AUC is probably the most meaningful for a detection task because it does not depend on any particular decision threshold—it measures how consistently the model ranks synthetic samples above genuine ones across the full operating range. In deployment, false negatives are typically the more consequential error type: a synthetic clip that slips through as genuine is a worse outcome, in most contexts, than a genuine clip that gets flagged. So recall on the synthetic class is worth watching separately, even though the overall numbers look good.



Classification Output: Genuine / Synthetic

Figure 1: Figure 1. End-to-end architecture of the CNN–BiLSTM deepfake voice detection system used in this work. Grey nodes represent input/output; blue nodes indicate signal processing stages; cyan nodes are the three convolutional blocks; violet nodes cover the recurrent and dense layers; and the green node is the final classification output.
Source: Authors

System Architecture and Deployment

Wrapping the trained model in a usable interface involved building a FastAPI backend with a single REST endpoint. The endpoint accepts audio file uploads in WAV, MP3, or FLAC format. Once a file arrives, the preprocessing steps run in sequence—resampling, normalisation, spectrogram extraction—and the resulting 128×87×1 tensor is passed to the model. The response is a JSON object with two fields: the predicted label (genuine or synthetic) and the raw sigmoid probability score. The model is loaded once at server start and stays in memory, which avoids the overhead of reloading it for every request. Scores above 0.5 map to synthetic; everything else maps to genuine. We also put together a lightweight browser-based front-end for users who are not comfortable calling APIs directly, though that part is fairly minimal—the backend is the real deliverable here.

VI. RESULTS AND DISCUSSION

Quantitative Performance

The numbers came out better than we expected. Table 1 summarises performance on the held-out test partition. Overall accuracy was 99.02%, with precision, recall, and F1-score all at 0.98. The AUC was 0.9998, which is close enough to 1.0 that we went back and re-ran the evaluation a second time just to be sure. It held up.

Table 1: Table 1. Classification performance on the FoR held-out test partition (n = 3,574 samples).

Metric	Value	Interpretation
Accuracy	99.02%	Overall correct classifications
Precision	0.98	Low false-positive rate
Recall	0.98	Low false-negative rate
F1-Score	0.98	Balanced precision–recall
AUC	0.9998	Near-perfect discriminative rank

Source: Authors

Baseline Comparison

To get a sense of how much the BiLSTM was actually contributing, we trained an identical network without the recurrent component—just the three convolutional blocks feeding directly into the dense output layer. As shown in Table 2, that model settled at 96.4% accuracy and an AUC of 0.9912. Adding the BiLSTM brought accuracy up by about 2.62 percentage points. That gap is consistent and shows up across the confusion matrix in a specific way: the CNN-only model made more false-negative errors in particular. This suggests that temporal sequence modelling is picking up on something genuinely useful, likely the subtle frame-to-frame irregularities in synthetic speech that do not manifest strongly in any individual spectrogram slice.

Table 2: Table 2. CNN-only baseline versus the full CNN–BiLSTM model under identical training and evaluation conditions.

Model	Parameters	Accuracy	AUC
CNN only (baseline)	≈0.8 M	96.40%	0.9912
CNN–BiLSTM (this work)	≈1.2 M	99.02%	0.9998
Improvement	—	+2.62 pp	+0.0086

Source: Authors

Spectrogram Analysis

Looking at spectrograms from correctly classified samples on both sides, some consistent visual patterns came up. Authentic speech tends to show smooth formant trajectories and the kind of slightly messy-looking transitions you get from natural coarticulation—it looks organic rather than regular. Synthetic spectrograms often appear just a little too uniform in the upper frequency range, almost as though the high-frequency detail has been smoothed or templated. There are also occasional faint discontinuities near segment boundaries, which is consistent with what is known about how certain neural vocoders handle the transitions between synthesis frames. These differences are subtle enough that a human looking at a spectrogram would likely miss them, but they appear reliably enough that the model can apparently use them.

Error Analysis

Across the full test set, the model made 35 errors in total. Thirty-three were false negatives—synthetic clips classified as genuine. Looking at those samples more carefully, they were disproportionately from the higher-quality TTS systems in the dataset: the ones that do a better job of mimicking natural prosody and avoiding the spectral discontinuities that lower-quality vocoders leave behind. A possible reason is that the model has learned to detect the artefacts of average-quality synthesis, not the characteristics of synthetic speech in general. As synthesis quality improves, those artefacts become harder to find—which is a known and persistent problem in this field, not unique to our approach (Korshunov and Marcel, 2019).

The two false positives were both genuine recordings with somewhat unusual acoustic properties. One had noticeable high-frequency background noise; the other was from a speaker with an unusually flat prosodic profile, almost monotone. Both ended up looking enough like synthetic artefacts in the spectrogram representation to confuse the classifier. With only two cases it is hard to draw strong conclusions, but it raises the question of whether atypical speaker styles could be a systematic weak point.

Limitations

A few things should be said plainly. First, the entire evaluation used clean, short-form audio. We do not know how well this holds up in noisy conditions, over compressed channels, or with clips longer than two seconds—and there are reasonable grounds to expect degraded performance in all three cases. Second, the FoR dataset does not include any diffusion-based vocoders, which represent roughly the current frontier of TTS quality. Whether the model would generalise to those systems is genuinely unknown. Third, the pipeline was not designed for streaming. It needs a complete file. These are the honest limitations of what we built, and they define where future work needs to go.

VI. CONCLUSION

We came into this work expecting decent results. The FoR benchmark is not trivially easy, but it is a clean, controlled dataset, and the CNN–BiLSTM combination is a reasonable architectural choice for this type of task. What we did not fully anticipate was how well the temporal modelling component would contribute—the gap between the CNN-only baseline and the full model was larger and more consistent than our initial estimates suggested.

The core finding is fairly straightforward: Mel spectrograms carry enough information to separate genuine from synthetic speech with high reliability, at least within this dataset's distribution, and modelling temporal dynamics with a bidirectional LSTM adds something real on top of what convolutional features provide alone. Whether that generalises beyond FoR is the question we cannot yet answer.

In practical terms, a detector of this kind could function as a pre-filter in voice authentication or content moderation pipelines—not as a final decision, but as a way to triage large volumes of audio and flag the subset most likely to be synthetic for human review. Even a

system that misses 1–2% of synthetic audio is substantially better than no automated check at all.

Going forward, three things seem worth pursuing. Making the model robust to real-world acoustic conditions, through augmented training, is the most immediately practical. Evaluating against diffusion-based vocoders would give a more honest measure of generalisation. And adapting the pipeline for streaming detection would unlock deployment contexts that the current architecture simply cannot support.

Acknowledgement

We thank the Department of Computational Intelligence at SRM Institute of Science and Technology, Chengalpattu, for access to computational resources and for guidance throughout this project. The Fake-or-Real dataset was made publicly available by its creators for non-commercial research use, and we are grateful for that.

REFERENCES

1. Alzantot, M., Wang, Z. and Srivastava, M. (2019) 'Deep residual neural networks for audio spoofing detection', in Proceedings of Interspeech 2019, Graz, pp. 1078–1082.
2. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. (2014) 'Generative adversarial nets', in Advances in Neural Information Processing Systems (NeurIPS), vol. 27, Montreal, pp. 2672–2680.
3. Hochreiter, S. and Schmidhuber, J. (1997) 'Long short-term memory', Neural Computation, 9(8), pp. 1735–1780. Kingma, D.P. and Ba, J. (2015) 'Adam: A method for stochastic optimization', in Proceedings of the 3rd International
4. Conference on Learning Representations (ICLR), San Diego, CA.
5. Korshunov, P. and Marcel, S. (2019) Deepfake Detection: Humans vs. Machines [online]. Available at: https://arxiv. org/abs/1910.13636 (Accessed: 1 January 2024).
6. LeCun, Y., Bengio, Y. and Hinton, G. (2015) 'Deep learning', Nature, 521(7553), pp. 436–444.
7. McFee, B., Raffel, C., Liang, D., Ellis, D., McVicar, M., Battenberg, E. and Nieto, O. (2015) 'Librosa: Audio and music signal analysis in Python', in Proceedings of the 14th Python in Science Conference, Austin, TX, pp. 18–25.
8. Tak, H., Jung, J., Patino, J., Kamble, M., Sahidullah, M. and Evans, N. (2022) 'RawBoost: A raw data boosting and smoothing method applied to automatic speaker verification anti-spoofing', in Proceedings of IEEE ICASSP 2022, Singapore, pp. 6382–6386.
9. Todisco, M., Wang, X., Vestman, V., Sahidullah, M., Delgado, H., Nautsch, A., Yamagishi, J., Evans, N., Kinnunen, T. and Lee, K.A. (2019) 'ASVspoof 2019: Future horizons in spoofed and fake audio detection', in Proceedings of Interspeech 2019, Graz, pp. 1008–1012.
10. van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. and Kavukcuoglu, K. (2016) WaveNet: A Generative Model for Raw Audio [online]. Available at: https://arxiv.org/ abs/1609.03499 (Accessed: 1 January 2024).
11. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and Polosukhin, I. (2017) 'Attention is all you need', in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, Long Beach, CA, pp. 5998–6008.
12. Wu, Z., Kinnunen, T., Evans, N., Yamagishi, J., Hanilci, C., Sahidullah, M. and Sizov, A. (2015) 'ASVspoof 2015: The first automatic speaker verification spoofing and countermeasures challenge', in Proceedings of Interspeech 2015, Dresden, pp. 2037–2041.