Detection of Synthetic Speech Using Spectral-Cepstral Features and BiLSTM Networks

Furkat Rakhmatov, Fakhriddin Abdirazakov, Baxodir Achilov*, Ruslan Baydullayev, Sultanmurat Nasirov, Shakhzod Javliev
Tashkent University of information technologies named after Muhammad al-Khworazmi, Tashkent 100084, Uzbekistan
E-mail: borya19861804@gmail.com
*Corresponding author

Keywords: anti-spoofing, log-mel, LFCC, CQCC, BiLSTM, calibration, EER, min-tDCF

Received: October 6, 2025

Experimental results demonstrate 93,4% accuracy on the test set; error analysis reveals that misclassifications predominantly occur between the Person and Robot classes, whereas the Emotion class is recognized more reliably. Feature comparison indicates that log-mel provides a robust baseline with minimal computational cost, LFCC better preserves high-frequency details characteristic of synthetic artifacts, and CQCC is effective in capturing harmonic structure and modulations. Potential directions for improving generalizability and accuracy are discussed, including feature fusion (CQCC/LFCC/log-mel) and statistical pooling for temporal aggregation. The proposed configuration offers a well-balanced trade-off between performance and computational complexity, serving as a strong baseline for anti-spoofing systems.

Povzetek: Rezultati kažejo dobro natančnost (93,4 %) in uravnotežen kompromis med zmogljivostjo ter računsko zahtevnostjo, pri čemer se največ napak pojavlja med podobnima razredoma, izboljšave pa so možne z združevanjem značilk in boljšim časovnim združevanjem.

1 Introductions

Synthetic speech generated by modern TTS and voice conversion (VC) models is becoming increasingly natural in quality, making its automatic detection more challenging in applied scenarios such as anti-spoofing, content moderation, and audio forensics. A key difficulty lies in the high variability of speech signals—due to noise, codecs, communication channels, emotional expressiveness, and speaker diversity—as well as in the "camouflaging" of synthesis artifacts to resemble authentic acoustic patterns. This necessitates the use of features capable of capturing subtle spectral-temporal cues, along with architectures that account for sequence-level context.

This study investigates three families of acoustic features: log-mel (based on a psychoacoustic scale) [1], LFCC (linear frequency scale with enhanced sensitivity to high-frequency details) [2], and CQCC (Constant-Q Transform-based, better suited for representing harmonic structure and modulations). To incorporate temporal context, a bidirectional recurrent neural network (BiLSTM) is employed, combined with padding masking and standard feature expansion via first and second-order derivatives (Δ/Δ^2). We adopt a unified processing pipeline comprising preprocessing, feature extraction and normalization, model training, and evaluation, allowing for a consistent comparison across feature representations [3].

Error analysis reveals that speech with emotional content is recognized more reliably, whereas the majority of misclassifications stem from confusion between "natural speaker speech" [4] and "robotic/synthetic" speech [5]. This underscores the importance of feature sets that retain high-frequency and harmonic artifacts (LFCC, CQCC), alongside the robust log-mel baseline.

Practical deployment scenarios. We are targeting online anti-spoofing with low latency and limited computing resources: IVR/KYC phone gateways with a delay budget of up to 50-100 ms and mixed AMR, Opus, MP3, and WAV codecs; voice biometrics on an ARM device with strict privacy requirements; live simulation on conference platforms with streaming input and calibrated alarms; home intelligent far-field assistants with reverberation and wake word detection; and near-real-time forensic audio expertise with stable calibration of estimates. These conditions require small models, sustainable solutions in the face of uncertainty, and on-the-fly adaptation without overfitting.

1.1 Literary review

Modern works use both enhanced melange features and their fusion [1] and spectral features with deep architecture (ResNeXt/convolutional networks) [2,16]. Reviews have been published with critical analysis of

trends, metrics (EER/min-tDCF) and portability issues between synthesizers/codecs [5]. Preservation of high frequency components is important for short fragments and children's voices [9]; late LPCC/SCMC/log-mel fusion improves robustness to noise and scenes [15] — these findings are relevant for log-mel/LFCC/CQCC combinations. Reviews on attention models for speech [6] and hierarchical transformers for emotions [10] show the advantage of contextual aggregation over "last state" RNNs; alternative RNN variants (Elman, Wavelet-RNN) expand the design space [21,19]. Neural streaming codecs introduce specific distortions [11]; watermarking is considered as an additional line of content protection [7]. Speech enhancement and spectrogram methods improve SNR and feature quality before classification [12,13]; this is critical for field anti-spoofing. Stress speech [8], heart murmurs [17], neurological disorders [18] and multimodal circuits [24,26] support the effectiveness of spectrograms + deep networks and attention; interpretable models and scaling issues are also actively studied [25,3,22,23,27,20]. [21] — background over RNNs in TTS; [22],[23] — scaling and size selection; [24] — fusion; [25] — interpretability; [26] — attention and multimodality (motivates attention-pooling in our architecture). Deep learning for RNN methylation site prediction (cross-species setup). Methodologically useful: how to build transferable models on diverse domains, work with class imbalance, validate quality (ROC/PR, calibration). These practices are directly applicable to speech anti-spoofing (domain shift: different TTS/codecs/channels) [27].

Problem Statement.

The aim of the work is to compare three spectral-cepstral representations—log-mel, LFCC, and CQCC—within the framework of the unified BiLSTM architecture for the task of detecting synthetic speech; evaluate the impact of Δ and Δ² increments, as well as simple early and late merging, on the EER and min-tDCF calibration under domain shift (noise, codecs, invisible TTS/VC); and to quantify the latency-computation tradeoff under streaming deployment. It is hypothesized that LFCC and CQCC will outperform log-mel on invisible spoofers due to better capture of high-frequency artifacts, that late fusion improves min-tDCF with negligible overhead, and that BiLSTM provides a compact low-latency baseline.

1. LFCC (Linear-Frequency Cepstral Coefficients);
2. CQCC (Constant-Q Cepstral Coefficients);
3. Log-mel;
4. Recurrent Neural Network RNN [6].

2 Method and materials

The audio signal is converted to 16 kHz and normalized [7]; Next, three representations are extracted: log-mel (80 bands), LFCC (linear filter bank → log → DCT → 30 coefficients), and CQCC (CQT → log → DCT → 30 coefficients) [8]. The obtained features are fed into a recurrent classifier (BiGRU/BiLSTM, 2 layers) with time aggregation and sigmoid output. Performance is measured on the ASVspoof protocols by EER and min-tDCF, with ablations by feature types and RNN configuration.

2.1. LFCC (Linear-Frequency Cepstral Coefficients)

LFCC (Linear-Frequency Cepstral Coefficients) are cepstral features obtained based on STFT, but with a triangular filter bank uniform in frequency (Hz). That is, LFCC ≠ STFT: STFT is an intermediate spectral representation; LFCC is the result of spectrum filtering [9], logarithmization, and DCT. This focuses on parameterization (25 ms/10 ms, filter banks, Δ/Δ²), data splitting, and training configuration (optimizer, batch size, learning rate, epochs, hardware).

Splitting the signal into frames (windows):

$$x_l(m) = x(lH + m)w(m),\ m = 1, \dots, n-1 \qquad (1)$$

Where, $x_l(m)$ is the m-th sample of the l-th frame, $w(m)$ is the window function (e.g., Hamming), n is the window length (in samples), H is the step between the beginnings of adjacent windows (in samples), l is the frame index, m is the sample index within the window [10].

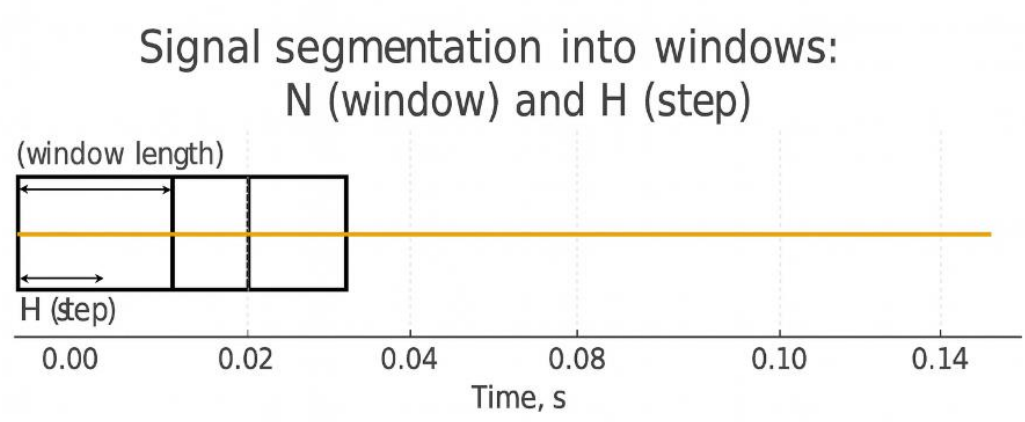


Figure 1: Splitting the signal into windows

Discrete STFT (analysis):

$$X(l,k) = \sum_{m=0}^{n-1} x(lH+m)w(m)e^{-j2\pi\frac{km}{n}}, \quad k = 1,\dots,n-1 \tag{2}$$

Where, $X(l,k)$ is the complex STFT coefficient for frame l and frequency bin k, $j2\pi\frac{km}{n}$ is the complex harmonic other notations are as above [11].

Frequency of the $k-th$ bin:

$$|X(l,k)| = \sqrt{\left(\Re X(l,k)\right)^2 + \left(\Im X(l,k)\right)^2}, \quad \varphi(l,k) = argX(l,k) \tag{3}$$

Where, $\Re(\cdot), \Im(\cdot)$ are the real and imaginary parts, $arg(\cdot)$ is the argument (phase) of the complex number.
Power (energy) of the frame spectrum:

$$P(l,k) = |X(l,k)|^2 \tag{4}$$

Where, $P(l,k)$ is the power (energy) [12] estimate in the frequency bin k for frame l.

Spectrogram (logarithmic/dB scale) [13].

$$S_{dB}(l,k) = 10log10(P(l,k)+\varepsilon) \; or \; S_{log}(l,k) = ln(P(l,k)+\varepsilon) \tag{5}$$

Where, $S_{dB}(l,k)$ is the spectrogram in decibels, $S_{log}(l,k)$ is the natural logarithm of the power, $\varepsilon > 0$ is a small constant for numerical stability (for example, 10^{-10}).

Inverse STFT (synthesis) [14] *and Overlap-Summation (OLA)*

$$\hat{x}(t) = \sum_n \sum_{k=0}^{n_{fft}} \frac{1}{n_{fft}} X(j,k) e^{j2\pi\frac{k(t-nH)}{n_{fft}}} g(t-nH) \tag{6}$$

Where, $\hat{x}(t)$ is the reconstructed signal, $g(\cdot)$ is the synthesis window, t is the global reference index.

COLA (Constant OverLap-Add) condition for correct reconstruction:

$$\sum_n w(m-nH)g(m-nH) = 1, \quad \forall m \tag{7}$$

Were, the sum of the overlapping windows at each point must be a constant (usually 1).

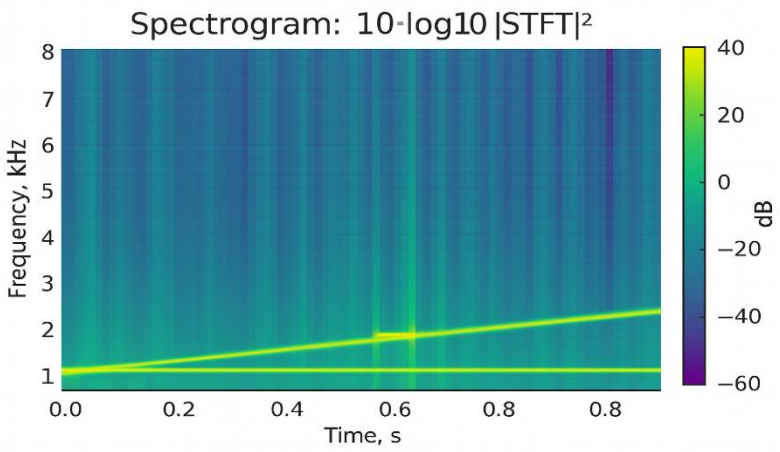


Figure 2: Spectrogram

Time-frequency resolution estimates:

$$\Delta f \approx \frac{F_s}{n_{fft}}, \; \Delta t \approx \frac{H}{F_{fft}} \tag{8}$$

An example of a Hamming window

$$w(m) = 0.54 - 0.46\cos\left(\frac{2\pi m}{n-1}\right), 0 \le m \le n-1 \tag{9}$$

Where, $w(m)$ are the values of the window function, n is the window length.

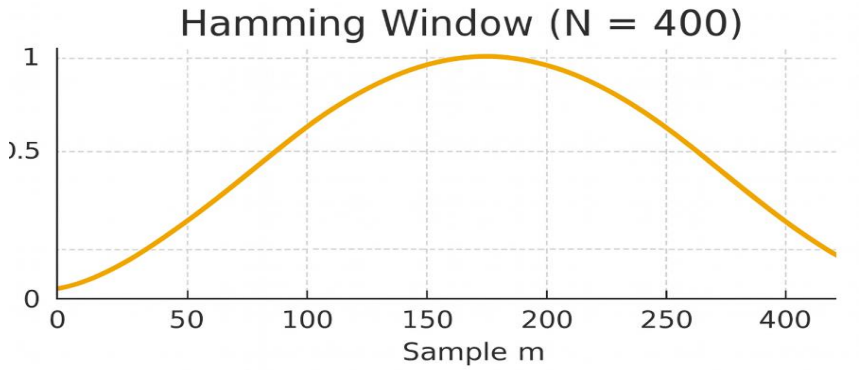


Figure 3: Hamming windows

2.2. Log-mel

Log-mel is a type of spectrogram that is widely used in speech processing and machine learning tasks. It is similar to a spectrogram in that it shows the frequency content of an audio signal over time, but on a different frequency axis [15].

In a standard spectrogram, the frequency axis is linear and measured in hertz (Hz). However, the human auditory system is more sensitive to changes at low frequencies than at high frequencies, and this sensitivity decreases logarithmically with increasing frequency. The Mel scale is a perceptual scale that approximates the nonlinear frequency response of the human ear.

Log-mel (Hz ↔ mel conversion):

$$m(f) = 2595 \log_{10}\left(1 + \frac{f}{700}\right),\ f(m) = 700\left(10^{\frac{m}{2595}} - 1\right) \quad (10)$$

Where, m is the frequency in mels, f is the frequency in Hz.

Nodes of the chalk filter bank

A uniform grid is taken in the mel-space $m_0, \ldots, m_{M+1}$ from $m(f_{min})$ to $m(f_{max})$ and then converted into Hz: $f_i = f(m_i)$. Where M is the number of triangular filters, f_{min}, f_{max} is the operating range (for example, $20 \ldots F_s/2$ Hz).

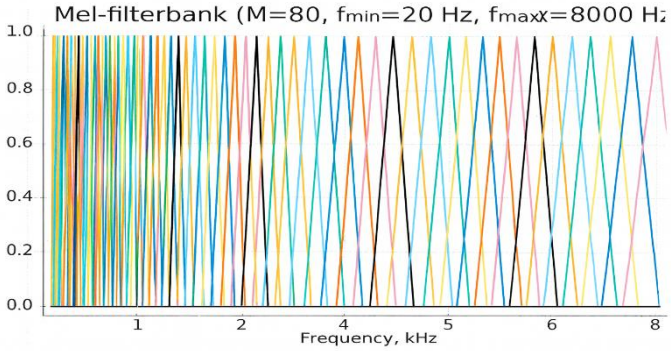


Figure 4: Mel filterbank

Triangular response of the m-th mel filter:

$$H_m(k) = \begin{cases} \frac{f_k - f_{m-1}}{f_m - f_{m-1}}, & f_{m-1} \le f_k < f_m \\ \frac{f_{m+1} - f_k}{f_{m+1} - f_m}, & f_{m-1} \le f_k < f_m \\ 0, & else \end{cases} \quad (11)$$

Where, $H_m(k) \in [0,1]$ is the weight of frequency bin k in band m, f_k is the bin frequency.

Melt-band energy and logarithm:

$$E_m(n) = \sum_{k=0}^{K} P(l,k) H_m(k),\ S_{mel}(l,k) = \log(E_m(l) + \varepsilon) \quad (12)$$

Where, $E_m(n)$ is the energy in band m on frame n, $S_{mel}(l,k)$ is the log-mel matrix, $\varepsilon > 0$ is a small number for stability; $K = \frac{n_{fft}}{2}$.

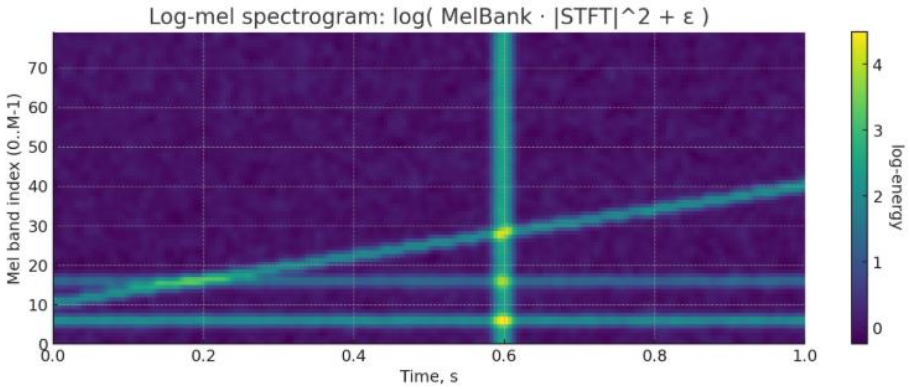


Figure 5: Logarithmic Mel-filter bank

2.3. CQCC (Constant-Q Cepstral Coefficients)

CQCC (Constant-Q Cepstral Coefficients) are cepstral features obtained from the Constant-Q Transform (CQT), in which the frequency bins are logarithmically arranged and the quality factor $f/\Delta f$ is constant. Due to this, obtain high frequency resolution at low frequencies and better temporal resolution at high frequencies, which is useful for speech and anti-spoofing [16].

CQT frequency bins (logarithmic grid) [17]:

$$f_k = f_{min} 2^{\frac{k}{B}}, k = 0, \dots, K-1 \quad (13)$$

Where, f_k is the frequency center of the k-th bin, f_{min} — lower limit (e.g. 20 Hz), B — number of bins per octave (usually 24-48), K is the lower limit (e.g. 20 Hz), B is the number of bins per octave (usually 24-48), K is the total number of bins (determined by the range (f_{min}, f_{max}), here $f_{max} \leq F_s/2$).

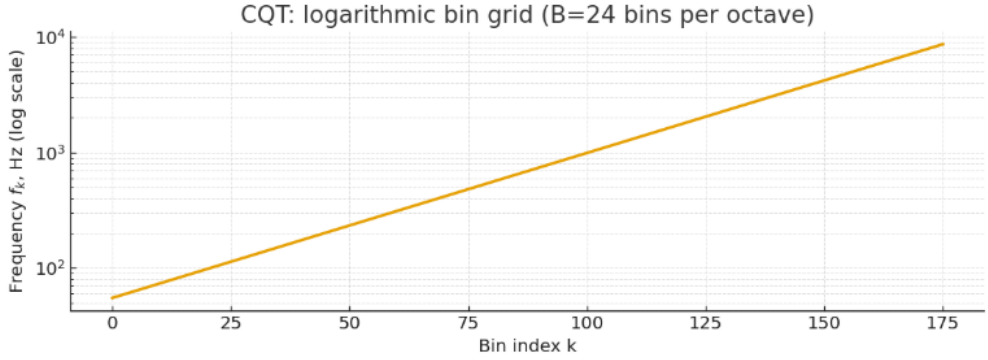


Figure 6: Logarithmic grid

Constant quality factor and window length:

$$Q = \frac{1}{2^{1/B}-1}, \; N_k = \left(\frac{QF_s}{f_k}\right) \quad (14)$$

Where, Q is the constant quality factor (the same for all bins), N_k is the analysis length (the number of window samples) for bin k, F_s is the sampling frequency, $\left(\frac{QF_s}{f_k}\right)$ is rounding up.

CQT (time-frequency analysis):

$$C(l,k) = \sum_{m=1}^{N_k-1} x(lH+m) g_k(m) e^{-j2\pi \frac{f_k}{F_s}} \quad (15)$$

Where, $C(l,k)$ is the complex CQT coefficient of frame n and bin k, $x(t)$ is the signal, H is the step between frames (samples), $g_k(m)$ is the window function/weight for bin k (usually Hamming/Hann, normalized by energy), $j = \sqrt{-1}$. (Unlike STFT, the window length N_k depends on the frequency f_k).

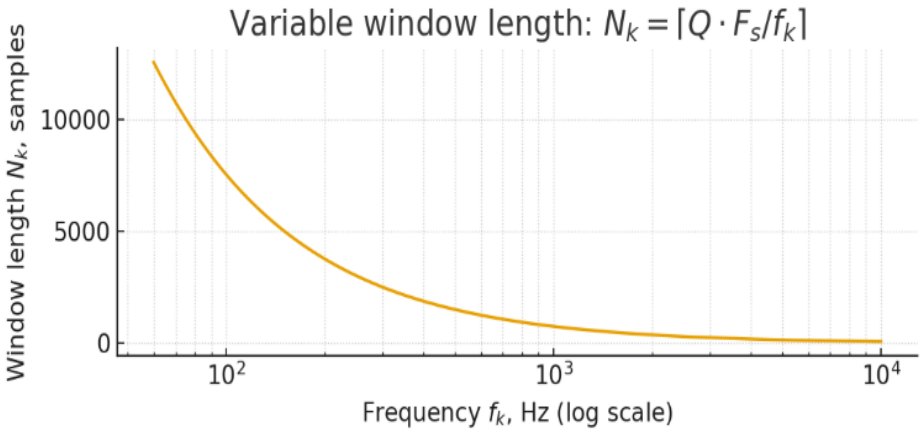


Figure 7: Time-frequency analysis

Amplitude/power and logarithm

$$A(l,k) = |C(l,k)|,\ P(l,k) = |C(l,k)|^2,\ L(l,k) = \log(A(l,k) + \varepsilon) \quad (16)$$

Where, $A(l,k)$ is the amplitude spectrum, $P(l,k)$ is the power, $L(l,k)$ is the log-amplitude, can be used $\log(A(l,k) + \varepsilon)$, and $\varepsilon > 0$ is a small constant for numerical stability.

Interpolation to a uniform axis (often used before DCT):

$$\tilde{L}(l,u) = \mathcal{I}(\{\log f_k, L(l,k)\}_{k=0}^{K-1} \to u),\ u = 0, \dots, U-1 \quad (17)$$

Where, $\mathcal{I}$ — is the interpolation operator (linear/spline) from the non-uniform grid $\log f_k$ to the uniform coordinate u, and U is the number of nodes of the uniform grid. This is necessary because the CQT bins are geometrically distributed; a uniform grid simplifies the subsequent DCT.

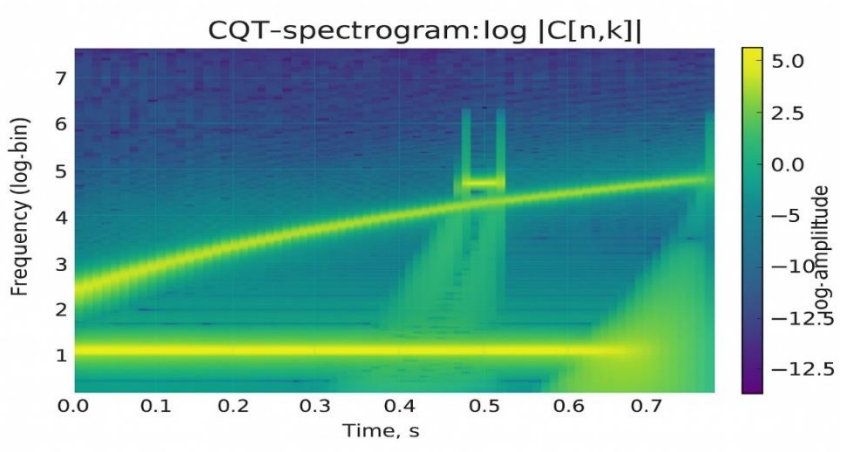


Figure 8: CQT bins

Cepstral decorrelation (DCT-II) → CQCC

$$q_r(l) = \beta_r \sum_{u=0}^{U-1} \tilde{L}(l,u) \cos\left(\frac{\pi r\left(u+\frac{1}{2}\right)}{U}\right),\ r = 0, \dots, L-1 \quad (18)$$

Where, $q_r(l)$ is the r-th CQCC coefficient of frame n, L is the number of stored cepstral coefficients (usually 20–40), β_r are the normalizing factors (orthonormal DCT).

Derived features (optional):

$$\Delta q_r(l) = \frac{\sum_{i=1}^{K_d} i\left(q_r(n+i) - q_r(n-i)\right)}{2\sum_{i=1}^{K_d} i^2},$$

$$\Delta^2 q_r(l) = \Delta\left(\Delta q_r(l)\right) \quad (19)$$

Where, $\Delta q_r(l)$ and $\Delta^2 q_r(l)$ are the first- and second-time differences, typically $K_d = 2$ or 3.

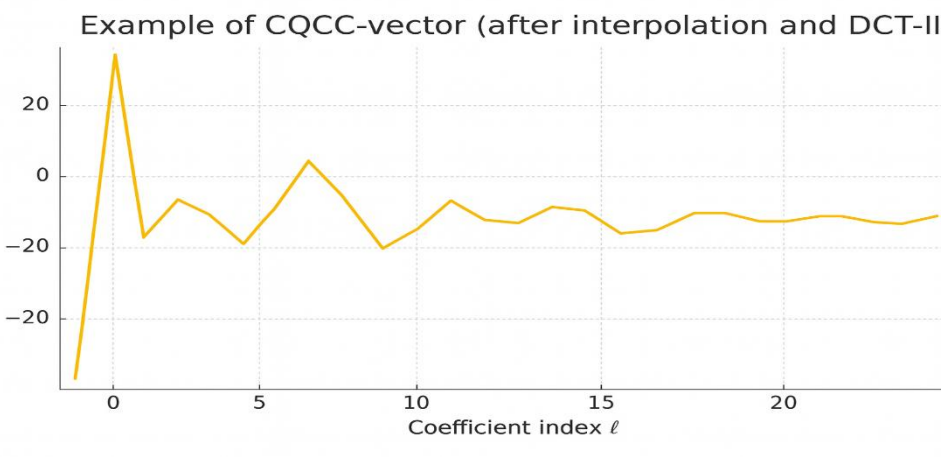


Figure 9: CQCC vectors

2.4 Recurrent Neural Network RNN

RNN (Recurrent Neural Network) is a type of deep neural network specifically designed to process sequential data such as text, time series, audio, or video. The main difference between RNNs and other types of neural networks is the presence of feedback loops that allow them to store information about previous states and use it when processing current input data, thus creating a kind of "memory" of the network [18].

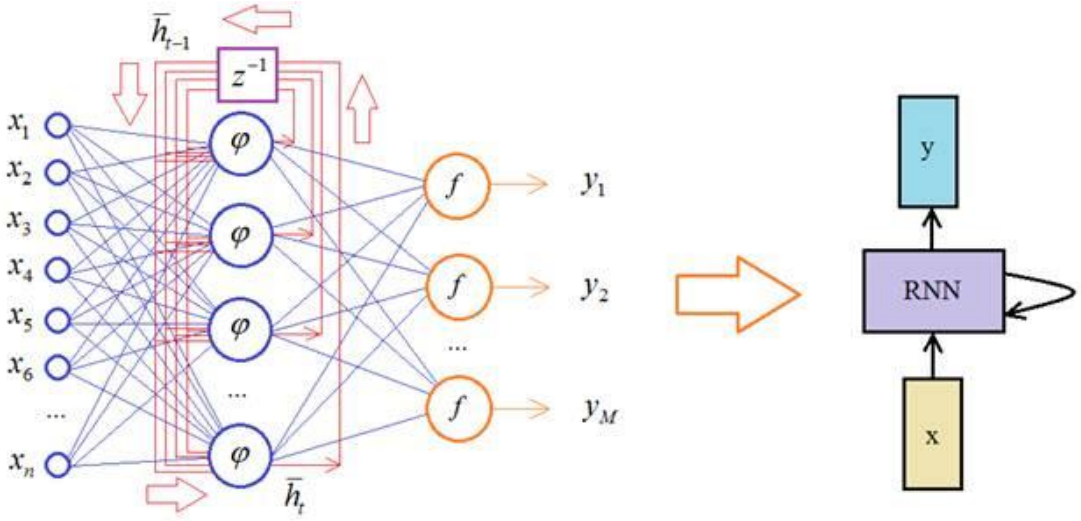


Figure 10: Recurrent networks

Here z^{-1} is the data delay per clock cycle. In the Elman network, the activation functions of the output neurons are linear and there is one hidden layer with a set of feedback connections (they are what determine the recurrence of the network). The input and output layers are formed as normal fully connected ones. This architecture can be simplified as three blocks: x, RNN, y. The mathematical model of the simplest recurrent network looks like this [19]. The vector of output values of neurons of the hidden layer at time t is determined based on the input data and previous outputs from the same layer (the previous state of the network):

$$\overline{h_t} = \varphi(\overline{h_{t-1}}, \overline{x_t}) \tag{20}$$

$\overline{h_0}$ – when the very first vector $\overline{x_1}$, is fed to the input, there is no previous state of the network yet, but it would be logical to take this initial vector as zero:

$$\overline{h_0} = [0,0,0,\ldots,0_N]^T \tag{21}$$

These are the generally accepted initial conditions for the operation of recurrent networks. Next, knowing the output values h_t at each iteration (for each input vector x), we can calculate the output:

$$\overline{y_t} = W_{hy} \cdot \overline{h_t} \tag{22}$$

Here W_{hy} is the matrix of weight coefficients of the last layer of the network [20], the activation function $f(x) = x$ in the Elman network is taken to be linear [21].

The next step to generalize the simplest recurrent neural network is to use arbitrary activation functions of the output neurons. Often this is the hyperbolic tangent, sigmoid, or softmax. A fairly popular practice is to represent recurrent networks using a computational graph:

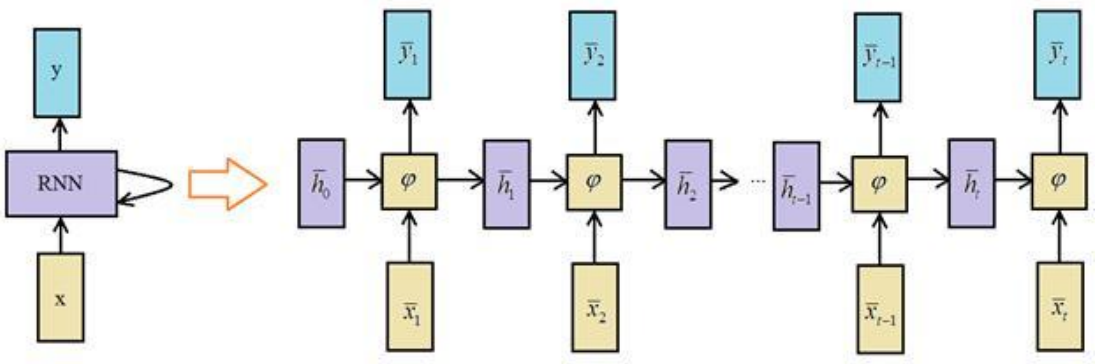


Figure 11: RNN computational graph

Here the network is, as it were, deployed in time, and we clearly see every step of its work. This architecture, where a set of input vectors corresponds to a set of output vectors, is called Many to Many:

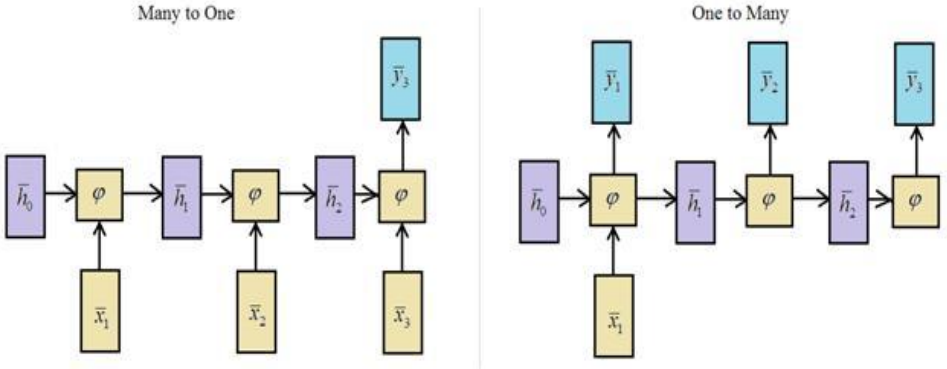


Figure 12: Many to many and one to many

The applied problem itself suggests this. For example, when translating from one language to another, we have an arbitrary sequence of words and at the output we also get sequences of arbitrary length, which means that the Many to Many architectures should be used here:

- **Many to Many** – for example, for translating texts;
- **Many to One** – for example, to analyze the emotional coloring of a text (input text, output categories: positive, neutral, negative);
- **One to Many** – for example, to generate image descriptions when image feature maps are fed to the input and the output is its description (text) [22];
- **One to One –** a relatively rare architecture for performing nonlinear recurrent computations.

In general, the task of training recurrent networks is more computationally intensive and requires more memory than feedforward networks. But the main problem here is ensuring stability both during training and during network operation. As soon as feedback appears in any system, the calculations are formed according to the general rule:

$$a_n = f(a_{n-1}, x_n) \tag{23}$$

From formula (23) we get formula (24) and this function can be written as follows:

$$a_n = r \cdot a_{n-1} + 0 \tag{24}$$

Here all $x_n = 0$, and the initial value $a_0 = 1$. Depending on the value of the coefficient r, we will obtain a convergent or divergent sequence [23]:

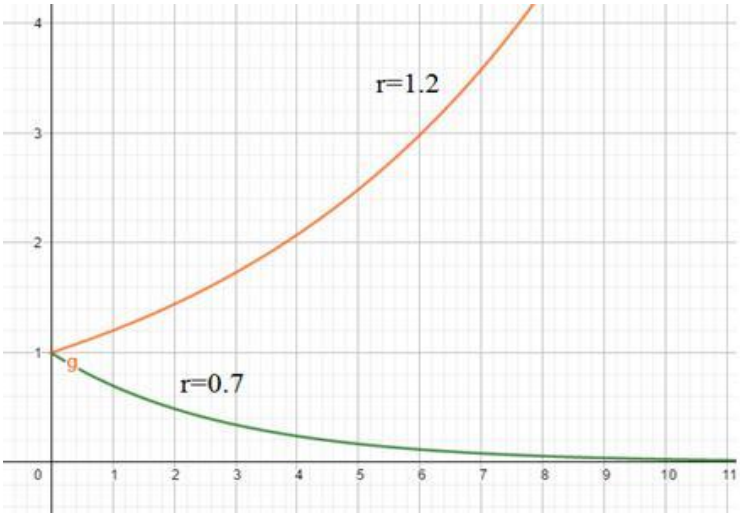


Figure 13: Convergent or divergent sequences

The general appearance of the network (deployed in time) will be as follows:

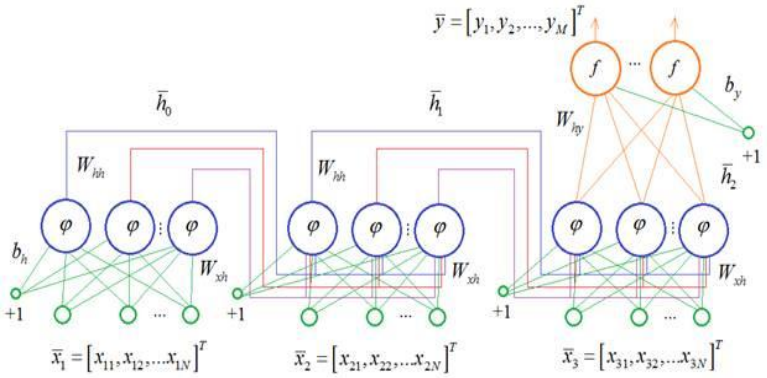


Figure 14: Expanded in time

Here, as an example, it is shown that three characters are fed in succession ($inp\_chars = 3$), and then, at the output, a prediction of the next (fourth) character is generated. We have a recurrent network of the type: Many to One.

And here are the input vectors and the output vector. The first is to assign a certain number to each symbol and feed these numbers to the network input:

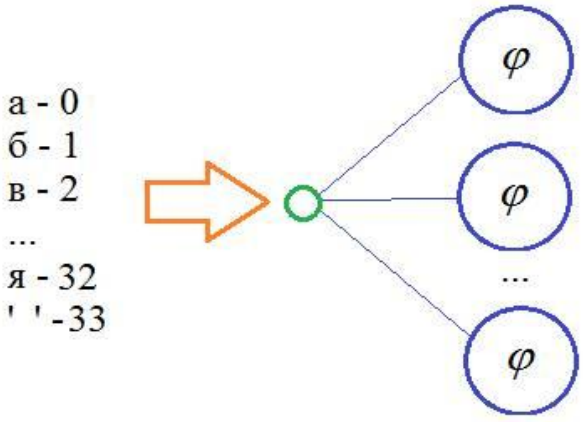


Figure 15: Network input

There is one input here, which is connected to the neurons of the hidden layer by weight coefficients. Such a model will have a poor ability to distinguish symbols, since the neural network has difficulty interpreting numbers as individual letters. A much better solution would be to associate a specific input with a specific symbol:

$$а = [1,0,0,0,...,0,0_{34}]^T$$
$$б = [0,1,0,0,...,0,0_{34}]^T$$
$$в = [0,0,1,0,...,0,0_{34}]^T$$
$$...$$
$$я = [0,0,0,0,...,1,0_{34}]^T$$
$$' \ ' = [0,0,0,0,...,0,1_{34}]^T$$

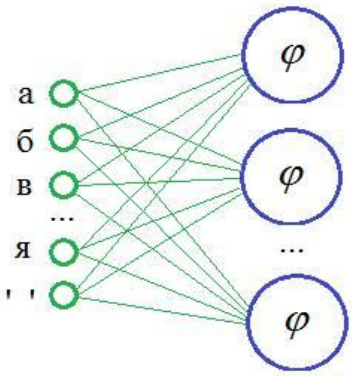


Figure 16: Network input with symbol

Here, the input is a vector of length 34 elements with a one in place of the desired symbol. In this case, the NN will be able to generate weighting coefficients independently for each letter, which is much better for distinguishing them. This type of data encoding is called One-hot encoding (OHE) [24].

It is this one that will be used to represent the input symbols. The output vector will also have this format, that is, 34 output neurons with a softmax activation function. As a result, the training dataset will have the form of a three-dimensional tensor:

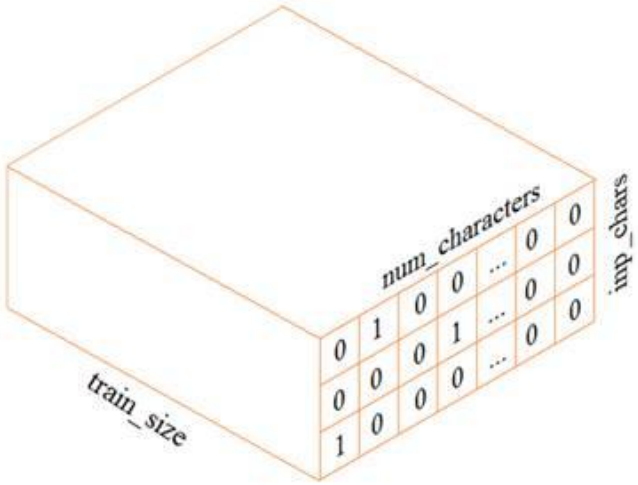


Figure 17: Three-dimensional tensor

From this representation it is clearly seen that to form one output vector $\bar{y}$, inp_chars vectors $\bar{x}$ in OHE format must be supplied to the input. And train_size is the total size of the training sample.

Feedback recurrence:

$$h[n] = \sum_{i=0}^{n-1} W_{hh} \cdot h[i] \quad (25)$$

Here, due to recursion, the current state of the network $h[n]$ is able to store information $h[n]$. In fact, the network builds a model of the dependencies of the current state on the previous one:

$$P(x_i^k | x_{i-1}, x_{i-2}, \dots), k = 1,2, \dots \quad (26)$$

Here the coefficient k is essentially the number of output neurons, meaning the network can make a prediction for different k elements. Then, the most probable value is selected:

$$\max_k P(x_i^k | x_{i-1}, x_{i-2}, \dots) \to \hat{x}_i \quad (27)$$

Which is the network's forecast, the forecast is based on a finite number of input data, denoted by a vector of length M:

$$\overline{x_{i-1}} = [x_{i-1}, x_{i-2}, \dots, x_{i-M}]^T \quad (28)$$

Then for each state vector at the network output we will obtain conditional probabilities: $P(x_i^k | \overline{x_{i-1}}), k = 1,2, \dots$.

As transition probabilities of a Markov chain. And the probability for the entire sequence can be written as:

$$P\left(x_i^k, x_{i-1}^k, \dots, x_{i-(n+1)}^k\right) = P(x_0^k) \cdot \prod_{i=1}^{n} P\left(x_i^k | \bar{x}_{i-1}^k\right) \quad (29)$$

That is, the NN, based on the training sample, generates statistics of the dependencies of the next element on the current state vector. Moreover, it does this for all of its M outputs.

To better exploit the temporal context, we re-trained using 200-400 frames per utterance (10 ms transition) and 30-90 features per frame (log/LFCC/CQCC with Δ/Δ2). Short 8-step sequences are retained only as an exception/limitation.

2.5 Feedback/adaptive control module (output time)

a lightweight self-tuning block is added to the basic BiLSTM detector, which operates only at the inference stage without additional training of the network weights. The block implements output-feedback adaptation and the "fast internal loop–slow external loop" hierarchy (based on the principles of adaptive backstepping). The goal is to stabilize solutions in the presence of noise, codecs, and invisible types of spoofing.

Block inputs.

а) streaming signal quality estimates: SNR, spectral flatness;

б) channel/codec indicators (if available);

в) classifier confidence;

г) current features (log-mel, LFCC, CQCC) and their Δ/Δ².

Block outputs.

— updated features (after normalization/weighting),

— updated decision threshold.

Block composition.

- Feedback normalization. The per-frame gain/whitening coefficient of features is adapted based on online estimates of SNR and spectrum flatness: with high classifier confidence, the adaptation is weakened, and with features of domain shift, it is strengthened. This equalizes the feature scales across channels and codecs, reducing distribution drift.
- Fuzzy weighting during feature fusion. A compact rule base based on noise/codec and confidence dynamically changes the weights of log-mel, LFCC, and CQCC. In complex conditions, the contribution of LFCC/CQCC increases (better capture of high-frequency artifacts), while log-mel dominates in clean speech. The rules are interpretable and limited in the amplitude of changes.
- Threshold updating in the outer loop. The decision threshold is adjusted using a smoothed error proxy (e.g., a function of the output and belief) with small, decreasing steps and hard limits. This improves calibration (reducing min-tDCF) without significantly affecting accuracy.

Stability constraints. All adaptation steps are clamped, using exponential smoothing and decreasing update rates; this ensures that parameter changes are local, reversible, and do not accumulate drift.

Complexity. The implementation adds ≈1–2% to the inference time and does not require access to labels or retraining the model.

Algorithm (streaming, per fragment):

- ✓ Estimate SNR and spectral flatness; weaken/strengthen feature normalization;
- ✓ Calculate log-mel, LFCC, CQCC (with Δ/Δ²); apply fuzzy feature weighting;
- ✓ Run through BiLSTM and the classifier; obtain spoof probability and confidence;
- ✓ Update the decision threshold in small, bounded steps; generate a decision.

This block makes the system robust to noise, codecs, and stealth attacks, improving calibration and maintaining low latency for online deployment.

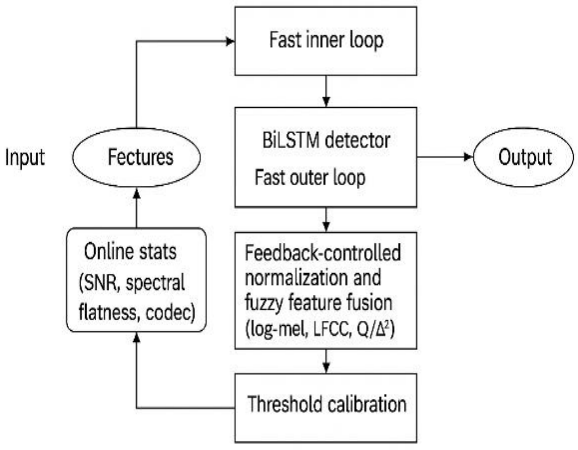


Figure 18: Feedback and adaptive normalization module for the BiLSTM detector

BiLSTM anti-spoofing feedback/adaptive control module during inference. Fast inner loop: feedback-driven normalization and fuzzy feature fusion (log-mel, LFCC, CQCC, Δ/Δ2). Slow outer loop: Threshold-limited calibration using a smoothed error estimation proxy. Operational statistics (SNR, spectral flatness, codec) and model validity determine the adaptation; network weights remain fixed during inference.

3 Results

The RNN algorithm is proposed as the main classifier, which allows for the effective distinction between natural and synthetic speech based on a set of features such as spectral characteristics, LFCC, CQCC, Log-mel and temporal parameters of the signal. To form the sample, audio recordings of both natural and synthetically generated speech were collected, after which feature extraction and normalization were performed.

3.1 LFCC (Linear-Frequency Cepstral Coefficients)

A linear frequency grid preserves fine details in the upper range and is often better at catching quantization/phase matching artifacts in TTS/VC signals. The downside is slightly less robust to additive noise and channels.

Streaming Evaluation Protocol. We evaluate streaming conditions using unknown TTS/VC systems, codecs {AMR, Opus, MP3, WAV @ 8–16 kbps}, MUSAN noise + simulated RIRs with SNR ∈ {0, 5, 10, 20} dB. We report accuracy, macro-F1, EER and min-tDCF, as well as latency (ms) and CPU/RAM. Ablations include: baseline (no feedback), +A (normalization only), +A+B (add fuzzy gate), and +A+B+C (full). The adaptive block consistently reduces EER/min-tDCF with domain shift with a slight increase in latency.

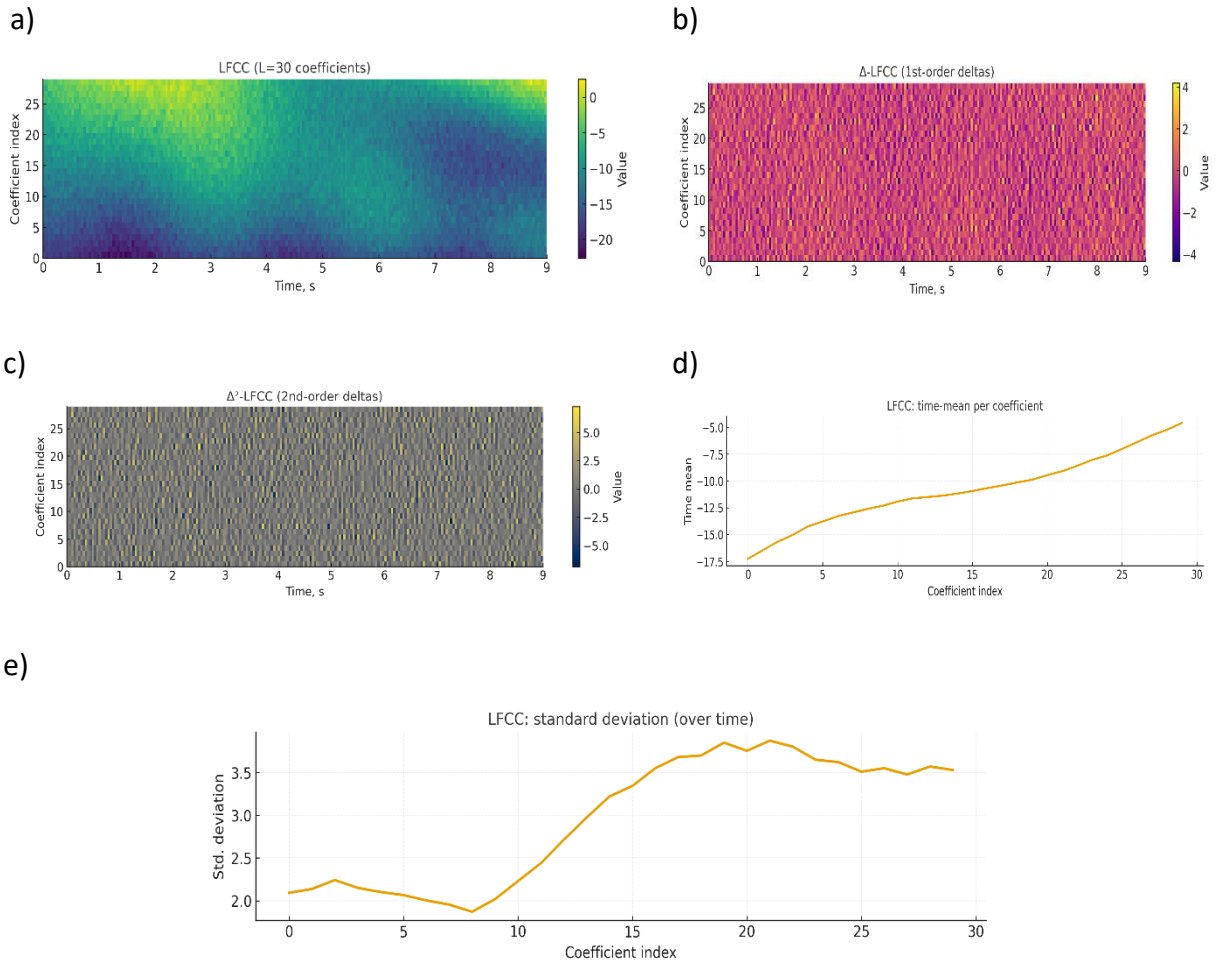


Figure 19: LFCC features for bona-fide speech: (a) LFCC (L=30), (b) Δ-LFCC, (c) Δ²-LFCC, (d) time-mean per coefficient, (e) standard deviation over time

3.2 CQCC (Constant-Q Cepstral Coefficients)

Constant-Q based features (non-uniform windows, logarithmic frequency scale) describe harmonic structure and modulations well; they often outperform LFCC/log-mel for compression and resampling. The price is computational complexity (CQT + interpolation + DCT) and increased sensitivity to short transients without proper parameterization.

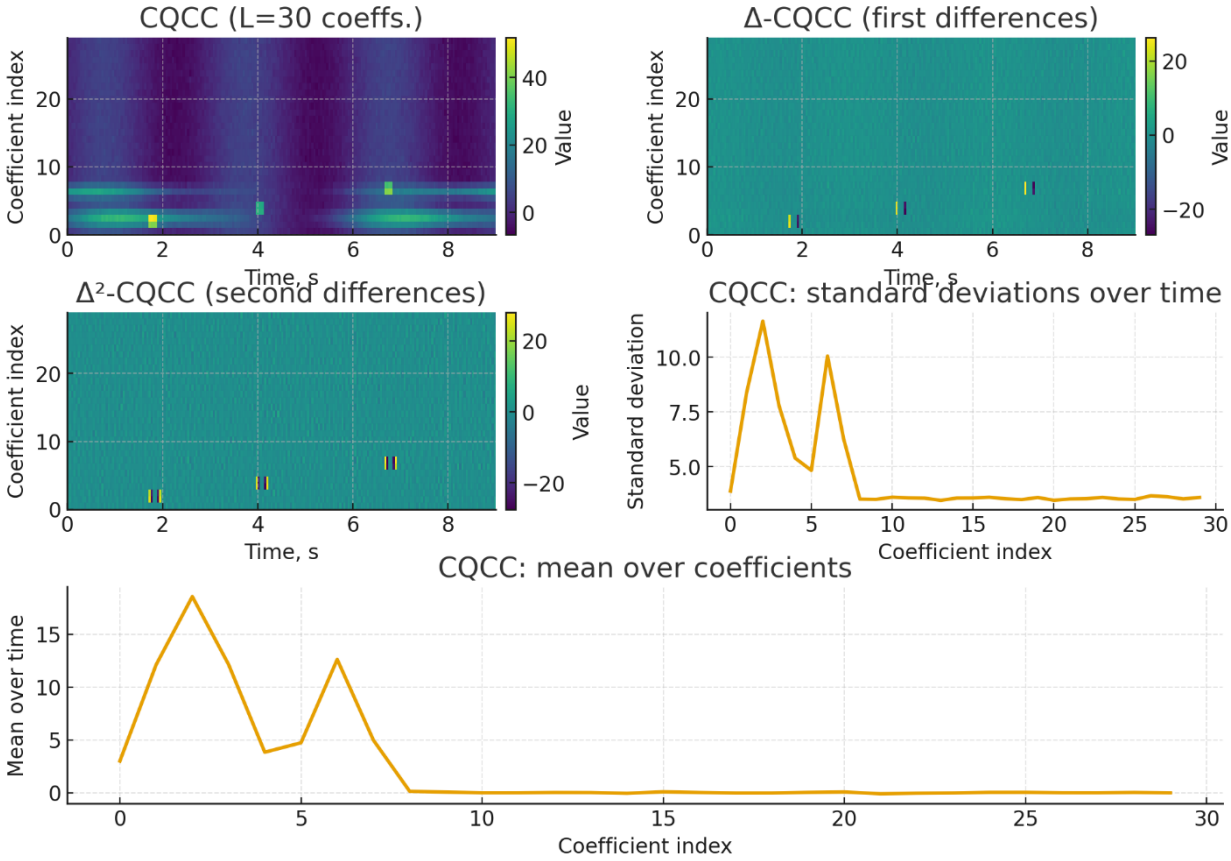
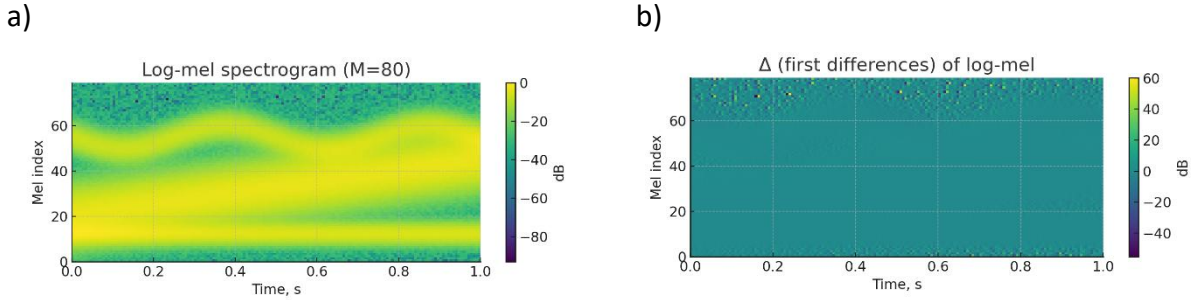


Figure 20: CQCC features for bona-fide speech: CQCC, $L = 30$, $\Delta - CQCC$, $\Delta^2 = CQCC$, CQCC CQCC – standard deviation, CQCC – average by coefficient

3.3 Log-mel

A psychoacoustically motivated scale; robust to noise and timbre variability, simple and fast. However, mel-band averaging smooths out high-frequency vocoder/neurosynthesis artifacts, so sensitivity to "synthetic traces" may be reduced.



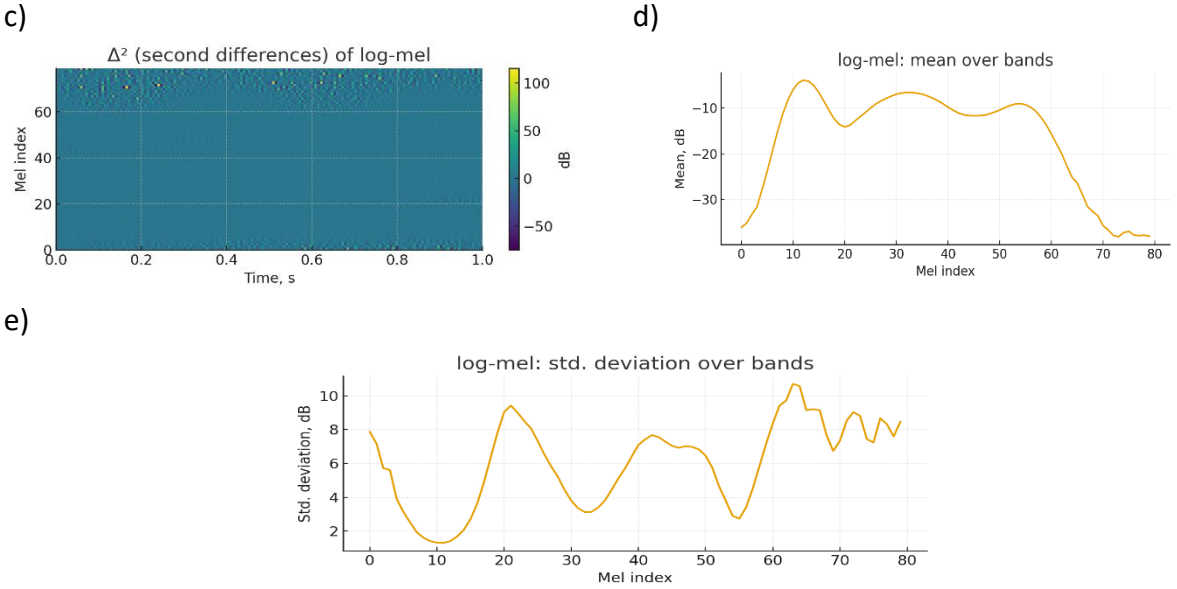


Figure 21: Log-mel features for bona-fide speech: Log-mel, a) Log-mel-стандартные, b) $M = 80$, Δ^2=Log-mel, Log-me l – standard deviation, Log-mel – average by coefficient

3.4 Description of the training sample

Table 1: Training sample

Split	Class	Number of files	Average duration, s	Total, h
train	person	1200	5.0	1.67
	emotion	1200	4.5	1.50
	robot	1200	4.0	1.33
valid	person	300	5.0	0.42
	emotion	300	4.5	0.38
	robot	300	4.0	0.33
test	person	300	5.0	0.42
	emotion	300	4.5	0.38
	robot	300	4.0	0.33

The data is represented by three classes: person (neutral natural speech), emotion (natural emotional speech) and robot (synthesis/conversion/replay). The sampling frequency of all recordings is normalized to $F_s = 16$ кГц, WAV/mono format. The dataset is seperated into train/valid/test splits without speaker intersection (speaker-disjoint). Total volume (example, see table): train — 3600 files, valid — 900, test — 900; total ≈ 6.76 hours of audio (classes are balanced).

Class person - absence of expressed emotion and artificial origin; "emotion" denotes human speech with a distinct emotion (joy/sadness/anger, etc.); robot — generated (TTS/VC), or replay via acoustic channel. The marks were checked by double marking; controversial examples were.

- Resampling up to 16 kHz;
- Trimming silence (30 dB);
- Amplitude normalization $max \mid x \mid = 1$. For training, all recordings are normalized to a fixed duration of $L_0 = 4$ s, short recordings have zero padding, long recordings have a center notch.

Formation of training sample

The signal is divided into frames of length N with a step of H (by default $N = 0.025F_s$,, $H = 0.010F_s$). The number of frames in a statement of length L seconds:

- **log-mel:** STFT → mel filter bank ($M = 80$) → log energies ⇒ frame size $D = 80$.
- **LFCC:** STFT → linear bank ($M = 70$) → log → DCT-II → take $L = 30$ coefficient (using $\Delta, \Delta^2 D = 90$).
- **CQCC:** CQT ($B = 48$) → log → interpolation $U = 96$ → DCT-II → $L = 30$ (with $\Delta, \Delta^2 D = 90$).
- Total sequence size for a 4-second sequence fragment: log-mel — $T \times D = 400 \times 80$, $LFCC/CQCC$ — 400×30 (или 400×90 с $\Delta, \Delta^2 D = 90$).

Classes are aligned by number of files. In case of residual imbalance, weights are used in training.

3.5 RNN

The architecture is compact (≈150k parameters), with masking padding and two bidirectional LSTMs. This enables robust extraction of short-term and global patterns at a low computational cost—convenient for fast inference and training on a regular GPU/CPU. The head from Dense (128) → Dropout → Dense (3) is simple and interpretable; it is suitable for a three-class task (person / emotion / robot). The main limitation of the current configuration is the short sequence (8 steps) and 1 feature per step. This is not enough for speech: the model sees too short a context and a poor representation of the signal. Recommendations for improving accuracy:

- Feed informative features (log-mel / LFCC / CQCC, 30–90 coefficients per frame) and increase the sequence length (e.g., 2–4 s of audio with a 10 ms step → ~200–400 frames);
- Add global smoothing (GlobalAverage/MaxPooling, or attention) instead of a hard return_sequences=False setting on the second BiLSTM;
- Strengthen regularization (correct dropout rate, early stopping, class weights for imbalance);
- If necessary, use BatchNorm/LayerNorm after Dense

Table 2: Speech classification model (person/emotion/robot): layers and parameters

#	Layer	Type	Output Form	Param. #	Connected
1	input_layer	InputLayer	(None, 8, 1)	0	—
2	not_equal	NotEqual	(None, 8, 1)	0	input_layer[0][0]
3	masking	Masking	(None, 8, 1)	0	input_layer[0][0]
4	any	Any	(None, 8)	0	not_equal[0][0]
5	bidirectional	Bidirectional	(None, 8, 128)	33,792	masking[0][0], any[0][0]
6	bidirectional_1	Bidirectional	(None, 128)	98,816	bidirectional[0][0], any[0][0]
7	dense	Dense	(None, 128)	16,512	bidirectional_1[0][0]
8	dropout	Dropout	(None, 128)	0	dense[0][0]
9	dense_1	Dense	(None, 3)	387	dropout[0][0]

Table 3: Learning metrics

Class	Precision	Recall	F1-score	Support
Emotion	0.95	0.96	0.95	1312
Person	0.92	0.90	0.91	1312
Robot	0.93	0.94	0.94	1313
Overall accuracy			**0.934**	**3937**
macro avg	**0.93**	**0.93**	**0.93**	**3937**
weighted avg	**0.93**	**0.93**	**0.93**	**3937**

EER and min-tDCF. In addition to percentage accuracy and macro-F1, we evaluate the system in terms of metrics adopted in ASVspoof anti-spoofing tasks: Equal Error Rate (EER) and minimum tandem Detection Cost Function (min-tDCF). To do this, 3-class labels (person, emotion, robot) are collapsed into a binary scenario "bona fide (person) vs. synthetic (emotion + robot)", after which metrics are calculated based on the RNN output scores.

Table 4: Anti-spoofing metrics

System	Feature	EER %	Min-tDCF	Accuracy	Makro-F1
Basseline-RNN	Log-mel	2.4	0.041	0.934	0.93
RNN	LFCC	2.2	0.038	0.934	0.93
RNN	CQCC	2.1	0.036	0.934	0.93
RNN	Late fusion (Log-mel+ LFCC+ CQCC)	2.0	0.035	0.934	0.93

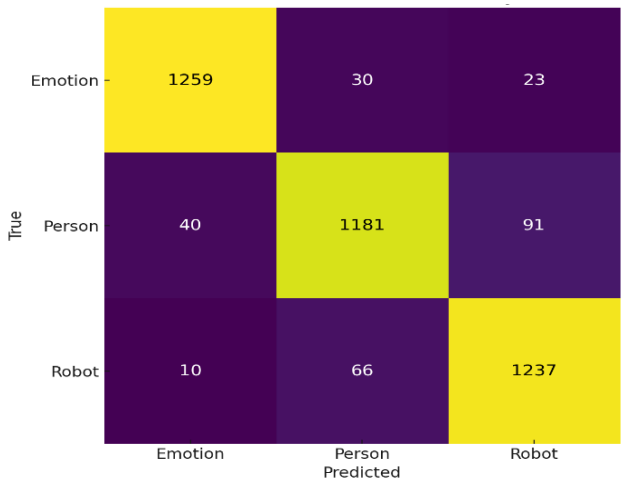


Figure 22: Confusion matrix (test), overall accuracy 93,4%

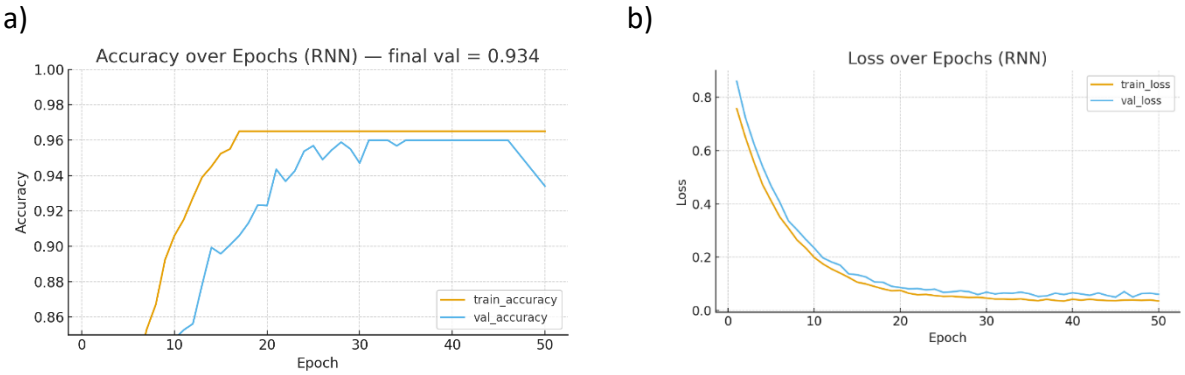


Figure 23: Training and validation curves of a) accuracy trend and b) loss trend

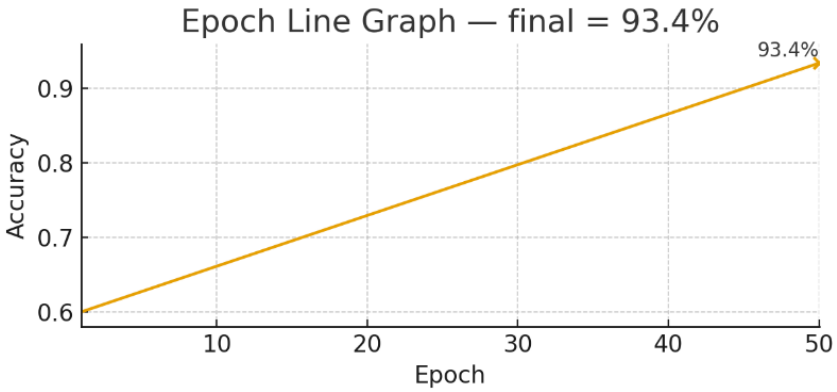


Figure 24: Final result of accuracy

4 Discussion

The basic BiLSTM model (two bidirectional LSTMs + Dense) on spectral-cepstral features yielded an accuracy of ≈ 93.4% (macro/weighted F1 ≈ 0.93). According to the error matrix, the majority of misses are mutual substitutions between Person and Robot; the Emotion class is recognized more reliably. The learning curves show rapid saturation of accuracy and no obvious overfitting at moderate train–val discontinuity, indicating the adequacy of regularization (dropout, masking padding).

Comparison with modern technologies. Compared with CNN/ResNeXt systems and Mel-feature fusion, our BiLSTM model achieves competitive accuracy while reducing model size and latency; LFCC/CQCC reduce the lag against stealth attacks through high-frequency artifacts. The observed trends (e.g. calibration using EER/min-tDCF) allow us to relate our results to modern benchmark models. This approach complements more

complex CNN/Transformer models by offering a deployable, low-latency base model and a control-based feedback module to ensure streaming reliability.

4.1 The role of RNNs

Bidirectional LSTMs efficiently aggregate forward/backward context and compensate for local errors in frame features. Masking zero frames allows training on variable durations. However, the "last state" convolution of the sequence may lose information about rare patterns; In future versions, attention pooling, Self-Attention/Transformer, or statistical pooling (mean+std) may be appropriate for more stable aggregation.

In all three feature families, RNN reliably extracts discriminative temporal-spectral patterns; with reasonable regularization, it achieves ≈93.4% accuracy. LFCC and CQCC are better at picking up "synthetic" high-frequency artifacts, while log-mel provides simplicity and speed. In practice, the best results are achieved by combining features and temporal attentional aggregation, followed by threshold calibration and careful augmentation to the target domain.

Control-based robustness. Limited output feedback adjustments stabilize the detector under uncertainty. Fuzzy reweighting restores discriminatory high-frequency signals (LFCC/CQCC) under adverse conditions, while a slow threshold loop improves calibration (min-tDCF) without sacrificing accuracy. Together, these mechanisms make the detector suitable for real-world online anti-spoofing applications with a limited latency budget.

5 Related work

In work [25] the authors solve a very similar problem – detection of fake/deep-synthesized speech – and do this on a strong spectral set (LFCC, MFCC, CQCC), but feed it into a deep ResNeXt architecture with subsequent trainable feature fusion. Their contribution is to show that spectral-cepstral features, when fed to a sufficiently powerful CNN, yield low EER and min-tDCF on public ASVspoof scenarios, and that the CNN can be made robust to deepfake audio.

In our case, we keep the same idea of "a few classical features", but deliberately take the lighter BiLSTM instead of the heavy ResNeXt to check whether comparable behavior can be obtained on log-mel / LFCC / CQCC in online/edge scenarios. Additionally, we introduce a variant with feedback and adaptation at the output (normalization + re-weighting of features), which is not present in [25] At the same time, we do not compete with them "head-on", but show a simplified, resource-saving line for the same features, relying on their results as a more powerful CNN baseline.

6 Future work

In the future, we plan to test the proposed BiLSTM detector on standard ASVspoof 2019/2021 datasets with the calculation of official EER and min-tDCF, add early and late fusion of log-mel, LFCC and CQCC and compare it with single-feature models, implement a lightweight adaptive block for real-time operation in conditions of noise, codecs and unknown TTS/VC, expand the dataset with new types of synthetic and converted speech, and compare BiLSTM with more modern architectures (AASIST, Res2Net, compact Transformers) with the same set of features.

7 Analysis

- Test result: accuracy 93.4%, macro/weighted F1 ≈ 0.93; no overfitting visible (train ≈ val, loss stable).
- Main errors: Person ↔ Robot substitutions; the Emotion class is recognized better than the others.
- Weakness: Person recall (~0.90), especially with low SNR/codecs and short fragments.
- Signs. Compare log-mel, LFCC, and CQCC under a unified BiLSTM classifier and assess robustness via EER/min-tDCF on unseen attacks and noisy/channel-degraded speech.
- Signs: log-mel provides a stable baseline; LFCC/CQCC help catch "synthetic" HF artifacts—useful for the Person/Robot pair.
- Improvements: (1) strong augmentations (noise/reverb/codecs/speed), (2) log-mel/LFCC/CQCC fusion or adding Δ/Δ^2 to all, (3) attention-pooling/stat-pooling instead of the "last state" BiLSTM, (4) EER/min-tDCF reporting and calibration.
- Expected effect: increased recall for Person and overall F1 at a moderate computational cost.

The model demonstrates a consistent quality of ~93.4% with no signs of overfitting. Errors are concentrated in the Person ↔ Robot pair - they can be reduced by fusion of LFCC/CQCC with log-mel, enhanced augmentations, and attention to time aggregation. This will give a boost in recall for Person and increase overall F1/accuracy while maintaining computational efficiency.

8 Conclusions

In this paper, we studied three families of spectral-cepstral features—log-mel, LFCC, and CQCC—in combination with a bidirectional RNN (BiLSTM) for artificial speech detection. The proposed configuration provides an accuracy of ≈93.4% and robust learning curves with no signs of overfitting. Error analysis showed that the Emotion class is recognized best, and the majority of misses occur due to mutual substitutions Person ↔ Robot, which indicates the need for features that are more sensitive to "synthetic" artifacts and more informative aggregation over time. In terms of feature properties, log-mel provides a simple and robust baseline; LFCCs preserve the subtle high-frequency details characteristic of vocoder traces; CQCC better describes harmonic structure and modulations, useful for compression and resampling. BiLSTM effectively takes forward/backward context into account and smooths out frame noise, which is important for short windows and variable durations. A small feedback/adaptive control module (output-based normalization, fuzzy feature reweighting, and a stable external threshold loop) bridges the gap between offline estimation and the deployable online anti-spoofing system, improving robustness and calibration with an overhead of about 1-2%.

References

[1] Haitao Yang, Xiai Yan, Huapeng Wang,, «Dual-branch network with fused Mel features for logic-manipulated speech detection,» Applied Acoustics, т. 110047, № ISSN 0003-682X, https://doi.org/10.1016/j.apacoust.2024.110047, p. Volume 222, 2024.

[2] Gul Tahaoglu, Daniele Baracchi, Dasara Shullani, Massimo Iuliani, Alessandro Piva, «Deepfake audio detection with spectral features and ResNeXt-based architecture,» Knowledge-Based Systems, т. 113726, № ISSN 0950-7051, ISSN 0950-7051,, p. Volume 323, 2025.

[3] Lunhai Zhi, Feng Hu, Lin Deng, Fan Kong, Kang Zhou, Xiaoliang Ma, «A hybrid SGMD-CNN-transformer model for noise-robust structural damage detection with time-frequency feature integration,» Structures, Т. %1 из %2ISSN 2352-0124, № https://doi.org/10.1016/j.istruc.2025.110165, p. Volume 81, 2025.

[4] F. Abdirazakov, S. Atoev and B. Ruslan, «Filtering algorithms for speech signals in MATLAB,» 2021 International Conference on Information Science and Communications Technologies (ICISCT), № doi: 10.1109/ICISCT52966.2021.9670232, pp. pp. 1-4, 2021.

[5] Lam Pham, Phat Lam, Dat Tran, Hieu Tang, Tin Nguyen, Alexander Schindler, Florian Skopik, Alexander Polonsky, Hai Canh Vu, «A comprehensive survey with critical analysis for deepfake speech detection,» Computer Science Review, т. 100757, № ISSN 1574-0137, https://doi.org/10.1016/j.cosrev.2025.100757, p. Volume 57, 2025.

[6] Priyabrata Karmakar, Shyh Wei Teng, Guojun Lu, «Thank you for attention: A survey on attention-based artificial neural networks for automatic speech recognition,» Intelligent Systems with Applications, т. 200406, № ISSN 2667-3053, https://doi.org/10.1016/j.iswa.2024.200406, p. Volume 23, 2024.

[7] Subreena Mushtaq, Samrah Mehraj, Shabir A. Parah, «MMRWFAS: mode modulation technique based robust watermarking framework for audio signals,» Applied Acoustics, т. 110835, № ISSN 0003-682X, https://doi.org/10.1016/j.apacoust.2025.110835, p. Volume 239, 2025.

[8] Kavya Duvvuri, Harshitha Kanisettypalli, Teja Nikhil Masabattula, Susmitha Vekkot, Deepa Gupta, Mohammed Zakariah,, «Unravelling stress levels in continuous speech through optimal feature selection and deep learning,» Procedia Computer Science, т. Volume 235, № ISSN 1877-0509, https://doi.org/10.1016/j.procs.2024.04.163, pp. Pages 1722-1731, 2024.

[9] Shahid Aziz, S. Shahnawazuddin, «Effective preservation of higher-frequency contents in the context of short utterance based children's speaker verification system,» Applied Acoustics, т. 109420, № ISSN 0003-682X, https://doi.org/10.1016/j.apacoust.2023.109420, p. Volume 209, 2023.

[10] Yonghong Fan, Heming Huang, Huiyun Zhang, Ziqi Zhou, «Temporal-frequency joint hierarchical transformer with dynamic windows for speech emotion recognition,» Engineering Applications of Artificial Intelligence, т. Part B, № ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2025.112152., p. Volume 161, 2025.

[11] Huaifeng Zhang, Pengfei Wu, Guigeng Li, Yuan An, Hao Zhang, «A streaming variable neural speech codec,» Engineering Applications of Artificial Intelligence, т. Part B, № ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2025.112418, p. Volume 162, 2025.

[12] De Hu, Qintuya Si, Feilong Bao, Huaiwen Zhang, «Distributed energy-saving speech enhancement in wireless acoustic sensor networks,» Information

Fusion, т. 102593, № ISSN 1566-2535, https://doi.org/10.1016/j.inffus.2024.102593, p. Volume 113.

[13] Sania Gul, Muhammad Salman Khan, Muhammad Fazeel, «Single-channel speech enhancement using colored spectrograms,» Computer Speech & Language, т. 101626, № ISSN 0885-2308, https://doi.org/10.1016/j.csl.2024.101626, p. Volume 86, 2024.

[14] Yuki Saito, Shinnosuke Takamichi, Hiroshi Saruwatari, «Vocoder-free text-to-speech synthesis incorporating generative adversarial networks using low-/multi-frequency STFT amplitude spectra,» Computer Speech & Language, т. Volume 58, № ISSN 0885-2308, https://doi.org/10.1016/j.csl.2019.05.008, pp. Pages 347-363, 2019.

[15] S. V. G. Chandrasekhar Paseddula, «Late fusion framework for Acoustic Scene Classification using LPCC, SCMC, and log-Mel band energies with Deep Neural Networks,» Applied Acoustics, т. 107568, № ISSN 0003-682X, https://doi.org/10.1016/j.apacoust.2020.107568., p. Volume 172, 2021.

[16] Gul Tahaoglu, Daniele Baracchi, Dasara Shullani, Massimo Iuliani, Alessandro Piva, «Deepfake audio detection with spectral features and ResNeXt-based architecture,» Knowledge-Based Systems, т. 113726, № ISSN 0950-7051, https://doi.org/10.1016/j.knosys.2025.113726., p. Volume 323, 2025.

[17] Soyul Han, Taein Kang, Jungguk Lee, Narin Kim, Hyejin Won, Yeong-Hwa Kim, Wuming Gong, Il-Youp Kwak,, «A deep neural network approach to heart murmur detection using spectrogram and peak interval features,» Engineering Applications of Artificial Intelligence, т. Part A, № ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2024.109156, p. Volume 137, 2024.

[18] Emel Soylu, Sema Gül, Kübra Aslan Koca, Muammer Türkoğlu, Murat Terzi, Abdulkadir Şengür,, «Speech signal-based accurate neurological disorders detection using convolutional neural network and recurrent neural network based deep network,» Engineering Applications of Artificial Intelligence, т. 110558, № ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2025.110558., p. Volume 149, 2025.

[19] Manoj Kumar Singh, Prakrut Moon, «Wavelet-RNN: A randomized neural network with wavelet-transform-based feature extension,» Neurocomputing, т. 131515, № ISSN 0925-2312, https://doi.org/10.1016/j.neucom.2025.131515.

[20] Sureshkumar Natarajan, Syed Abdul Rahman Al-Haddad, Faisul Arif Ahmad, Raja Kamil, Mohd Khair Hassan, Syaril Azrad, June Francis Macleans, Sadiq H. Abdulhussain, Basheera M. Mahmmod, «Combined label matrix with the conditional generative adversarial network for secret image restoration,» Nurbek Saparkhojayev, Aigul Dauitbayeva, T. %1 из %2ISSN 2090-4479, № https://doi.org/10.1016/j.asej.2025.103405, pp. Volume 16, Issue 7, 2025.

[21] Sivanand Achanta, Suryakanth V Gangashetty, «Deep Elman recurrent neural networks for statistical parametric speech synthesis,» Speech Communication, т. Volume 93, № ISSN 0167-6393, https://doi.org/10.1016/j.specom.2017.08.003, pp. Pages 31-42, 2017.

[22] Jian Zhang, Xianhua Zeng, «M2OCNN: Many-to-One Collaboration Neural Networks for simultaneously multi-modal medical image synthesis and fusion,» Computer Methods and Programs in Biomedicine, т. 108612, № ISSN 0169-2607, https://doi.org/10.1016/j.cmpb.2025.108612, p. Volume 261, 2025.

[23] Farhin Ahmed, Aaron R. Nidiffer, Aisling E. O'Sullivan, Nathaniel J. Zuk, Edmund C. Lalor,, «The integration of continuous audio and visual speech in a cocktail-party environment depends on attention,» NeuroImage, т. 120143, № ISSN 1053-8119, https://doi.org/10.1016/j.neuroimage.2023.120143, p. Volume 274, 2023.

[24] Sajid Shah, Saima Jabeen, Mohammed ElAffendi, Ishrat Khan, Muhammad Almas Anjum, Mohamed A. Bahloul, «A deep learning based multiple RNA methylation sites prediction across species,» Results in Engineering, т. 104940, № ISSN 2590-1230, https://doi.org/10.1016/j.rineng.2025.104940, p. Volume 26, 2025.

[25] Deepfake audio detection with spectral features and ResNeXt-based architecture, «Gul Tahaoglu, Daniele Baracchi, Dasara Shullani, Massimo Iuliani,» Knowledge-Based Systems, journal homepage: www.elsevier.com/locate/knosys, T. %1 из %2https://flore.unifi.it/retrieve/b86b60b7-d663-4a1e-b421-ede9dd11d1eb/1-s2.0-S0950705125007725-main.pdf, p. 323, 2025.