electronics

MDPI

Article

Two-Tier Feature Extraction with Metaheuristics-Based Automated Forensic Speaker Verification Model

Gaurav *, Saurabh Bhardwaj and Ravinder Agarwal

Electrical and Instrumentation Engineering Department, Thapar Institute of Engineering and Technology, Patiala 147004, India; saurabh.bhardwaj@thapar.edu (S.B.)
* Correspondence: ggaurav_phd16@thapar.edu

Abstract: While speaker verification represents a critically important application of speaker recognition, it is also the most challenging and least well-understood application. Robust feature extraction plays an integral role in enhancing the efficiency of forensic speaker verification. Although the speech signal is a continuous one-dimensional time series, most recent models depend on recurrent neural network (RNN) or convolutional neural network (CNN) models, which are not able to exhaustively represent human speech, thus opening themselves up to speech forgery. As a result, to accurately simulate human speech and to further ensure speaker authenticity, we must establish a reliable technique. This research article presents a Two-Tier Feature Extraction with Metaheuristics-Based Automated Forensic Speaker Verification (TTFEM-AFSV) model, which aims to overcome the limitations of the previous models. The TTFEM-AFSV model focuses on verifying speakers in forensic applications by exploiting the average median filtering (AMF) technique to discard the noise in speech signals. Subsequently, the MFCC and spectrograms are considered as the inputs to the deep convolutional neural network-based Inception v3 model, and the Ant Lion Optimizer (ALO) algorithm is utilized to fine-tune the hyperparameters related to the Inception v3 model. Finally, a long short-term memory with a recurrent neural network (LSTM-RNN) mechanism is employed as a classifier for automated speaker recognition. The performance validation of the TTFEM-AFSV model was tested in a series of experiments. Comparative study revealed the significantly improved performance of the TTFEM-AFSV model over recent approaches.

Keywords: automated speaker recognition; deep learning; ant lion optimizer; feature extraction; spectrograms; speech signals

Citation: Gaurav; Bhardwaj, S.; Agarwal, R. Two-Tier Feature Extraction with Metaheuristics-Based Automated Forensic Speaker Verification Model. *Electronics* **2023**, *12*, 2342. https://doi.org/10.3390/electronics12102342

Academic Editor: Valeri Mladenov

Received: 18 March 2023
Revised: 19 May 2023
Accepted: 20 May 2023
Published: 22 May 2023

1. Introduction

Speaker recognition is a method for determining the identity of a person from the sound of their voice. Every sound is distinct due to differences in the size of the larynx, the shape of the vocal tract, and other parts of the voice production organs [1]. This technique is complex, because the test speaker does not state their identity. The process needs to carry out a 1: N classification, where N refers to the number of speakers enrolled. Speaker verification is the process of verifying the identity whether a speaker is who they claim to be; as false claims are regarded to be unknown, this is commonly known as open-set recognition [2]. Generally, if the speaker's voice is considered to be coming from a known speaker, this is called closed-set recognition. Human speech signals are a substantial medium of transmission that contains rich information, namely emotional traits, accent, gender, and so on. These unique features enable researchers to recognize speakers through voiceprint identification [3]. The collected speaker utterance is fed into the deep learning network for training. When using an identification technique, the speaker recognition method matches the extracted features of the speaker with those in the model library [4]. Next, the speaker with the maximum likelihood of having delivered the utterance is identified as the target speaker [5].

Automated speaker recognition techniques are able to identify a person on the basis of their voice signal. Automated speaker recognition is deployed in a variety of applications, namely to control access to services like voice mail, telephone banking, data network voice dialing, database services, remote access to computers, information services, telephone shopping, and law enforcement/monitoring, namely security control of web services and for private data areas, forensics for crime investigation, remote time logging, and prison call observation [6]. It is essential to extract features from every frame containing key features of the speech signal for the speaker verification system [7]. The linear prediction cepstral coefficient (LPCC), perceptual linear predictive coefficient (PLPC), and Mel frequency cepstral coefficient (MFCC) are different feature extraction techniques utilized in speaker verification systems. The MFCC is extensively employed as the feature extraction technique in current speaker verification systems, and it achieves higher efficiency under clean conditions [8]. However, the efficiency of the MFCC feature drops considerably in situations characterized by reverberation and noise [9]. The major problem with current techniques is that the size of every input sample needs to be massive enough to be able to correctly perform training and identify the speaker, thus increasing the resources required [10]. Additionally, they are not able to work with optimum performance in noisy environments. In one study, a strong speaker identification technique was developed using a hybrid mechanism for automatically recognizing the speaker efficiently from speech signals [11].

In this article, a Two-Tier Feature Extraction with Metaheuristics-Based Automated Forensic Speaker Verification (TTFEM-AFSV) model is presented. The proposed TTFEM-AFSV model uses the average median filtering (AMF) technique to discard the noise present in speech signals. Subsequently, the MFCC and spectrograms are considered as the input to the deep convolutional neural network-based Inception v3 model. Moreover, the ant lion optimizer (ALO) algorithm is utilized to fine-tune the hyperparameters related to the Inception v3 model. Finally, a long short-term memory with a recurrent neural network (LSTM-RNN) mechanism is employed as a classifier for automated ASR. The performance validation of the TTFEM-AFSV model is tested under a series of experiments.

2. Literature Review

Al-Ali et al. [12] examined the efficiency of integrating the MFCC feature. This was extracted from the DWT model, with or without feature warping, to improve the current identity-vector (a-vector)-based speaker verification efficiency in the presence of reverberation and noise. The technique was utilized to enhance the efficiency of forensic speaker verification (FSR) and to prepare legal evidence in court. Huang et al. [13] developed a multiple-layer hybrid fuzzy SVM (MLHF-SVM) method that involved three layers: classification, pre-classification, and feature extraction layers. The MLHF-SVM method resolved the abovementioned issue by using FCM-based identification data of human and multilayer SVM classifiers. Additionally, to address the shortcoming whereby FCM tends to become trapped in local minima, an improved natural exponential inertia weight PSO (IEPSO) approach was developed and incorporated with FCM for optimization.

Swain et al. [14] developed a DNN method based on a 2D-CNN and GRU for speaker recognition. In the network architecture, the convolution layer is utilized to reduce dimensionality in frequency and time domains and voiceprint feature extraction, allowing fast GRU layer calculation. Furthermore, the stacked GRU recurrent network layer learns a speaker's acoustic features. Farhatullah et al. [15] successfully recognized a recording of a telephone conversation compared to unexpected sound recordings using an SVM model. Saleem et al. [16] introduced a novel FSR methodology that was dependent on extracting language and accent data from short words. The researchers used distinct baseline and DL methodologies to automate these processes. The newly developed CNN-based models, such as GMM-CNN and VGGVox, were applied, and both models utilized speech spectrograms. In the case of DNN, an x-vector model was used that was dependent on DNN embedding.

Khan, F. et al. [17] highlighted how crucial the filter feature selection is. It was shown that the most significant sound characteristics were determined using the method of filter feature selection and logistic regression, random forest, and K-nearest neighbor (KNN). Snyder et al. [18] demonstrated that a highly successful method for enhancing the performance of DNN embedding systems is to artificially add noise and reverberation to the training data. The Cantonese section of the NIST SRE 2016 evaluation (SRE16) and speakers in the wild were used as the basis for the implementations in this research [19]. This inspired us to try with speaker verification assessments using a naturally generated database under uncontrolled circumstances.

Devi et al. [20] described a new hybrid mechanism for ASR with a speech signal-based ANN model. The MFCC has been successfully employed for extracting features. The extracted features generate the input sample, and the Self-Organizing Feature Map (SOFM) is used to decrease the dimensions. Lastly, with the reduced input sample, recognition can be achieved using the MLP model with Bayesian regularization. Teixeira et al. [21] showed that speaker embedding can be extracted while keeping the speaker's voice and the service provider model private by using Secure Multiparty Computation. Furthermore, we offer the potential to attain reasonable trade-offs between computation and security costs. This study is complementary to those showing how verification might be privately implemented and thus can be regarded as another step toward entirely private ASR. The previous approaches reported were tested on a controlled database with some assumptions; we used the database created in an uncontrolled environment and still managed to achieve reasonable accuracy.

3. The Proposed Model

In this article, a new TTFEM-AFSV model is developed to verify the identity of speakers in forensic applications. The combined development sets include 400 samples from five speakers, and the speech is sampled at a rate of 16 kHz. The development set is divided into 20% for validation and 80% for training. Each set of speech fragments lasts between 3 and 5 s. The TTFEM-AFSV model initially employed the AMF technique to discard the noise present in the speech signals. Next, the MFCC and spectrograms are considered as input to the Inception v3 model. Then, the ALO algorithm is utilized to fine-tune the hyperparameters related to the Inception v3 model. Finally, the LSTM-RNN model is exploited as a classifier for automated ASR. Figure 1 depicts the overall process of the TTFEM-AFSV approach.

3.1. Pre-Processing

The median filter (MF) retains the sharpness of an image and eliminates the noise. All of the pixels are substituted with median values from the neighboring pixel. This filter utilizes a 3 × 3 window [22]. It is the best filter among the traditional filters for eliminating speckle noises. The steps followed to create the MF are shown in Algorithm 1. Spatial processing to retain the edge details and to remove non-impulsive noises through the adaptive MF plays a crucial part. The AMF preserves the smaller structure in the image and edge. In the AMF, the window size differs for every pixel.

Algorithm 1: Pseudocode of AMF

(1) Consider "A" input matrix with N columns and M rows.
(2) Create a matrix with N + 2 columns and M + 2 rows by adding a zero to the side of the input matrix
(3) Take a mask of size 3 × 3.
(4) Place the mask on the initial component, viz., the first column and row of matrix "A".
(5) Select each element listed with the mask and arrange them in ascending sequence.
(6) Take the median value (center component) from the sorted array and substitute the component A(1, 1) with the median values
(7) Slide the mask to the following component.
(8) Reiterate the 4 to 7 steps until each matrix "A" element is substituted with the respective median values.

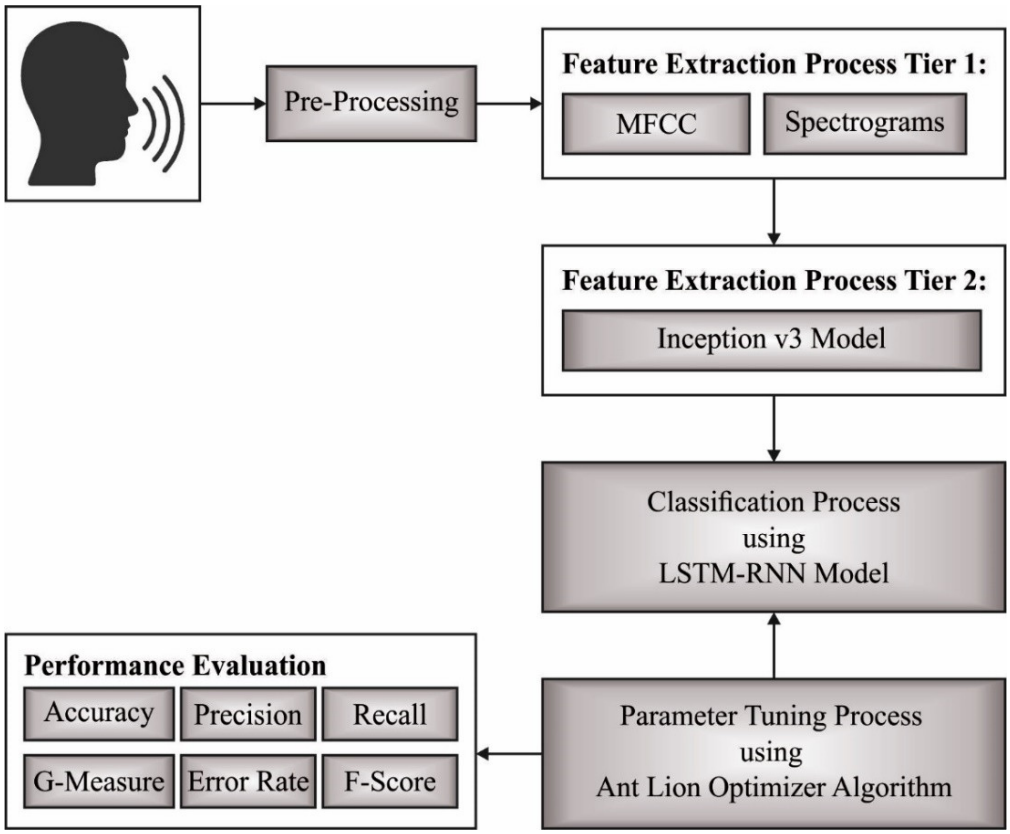

Figure 1. Overall process of the TTFEM-AFSV approach.

3.2. MFCC and Spectrograms

Spectrograms and MFCC are the two speech features utilized in this work. MFCC depends on a cepstral analysis of speech signals and has been extensively applied in different speech applications [23]. The cepstral analysis depends on the segmentation of the speech sample into frames with a length of 25 ms, and with a frameshift of 10 ms between the neighboring instances. All of the frames are multiplied by the Hamming window, and later, Fast Fourier Transform (FFT) is employed. The FFT response is filtered through a triangular Mel-filter bank. Then, the filter bank response is processed with MFCC, and Discrete Cosine Transform features are attained. MFCC is normalized with mean and variance in order to obtain the zero mean and unit variance features of the spectrogram on the basis of the frequency domain analysis of the speech signal. This is calculated in a sliding window manner with a Hamming window with a step size of 10 m and a length of 25 ms. Normalized mean and variance are employed on the spectrogram to obtain the zero mean and unit variance features.

3.3. Inception v3-Based Feature Extraction

The presented method uses the MFCC spectrogram as the input of CNN; the spectrogram is a two-dimensional signal and constitutes the identity data of the speaker. Simultaneously, CNN provides translation invariance in time and space. Hence, the voiceprint features in the spectrogram space can be obtained without terminating the time series. As a result, in this study, the MFCC and spectrogram are proposed as the input for the Inception_v3 module [24]. Inception_v3 is an alternative to the presented Google Net image recognition Inception structure. The Inception modules generally contain convolution and maximal grouping layers of dissimilar size. A channel is added to the network output of the preceding layer during convolution, and a non-linear combination is subsequently implemented. Overfitting can be evaded, and the network adaptability and expression on

various scales are enhanced. Inception_v3 is a Keras-developed network architecture that is trained in ImageNet. Figure 2 illustrates the process of sample feature extraction.

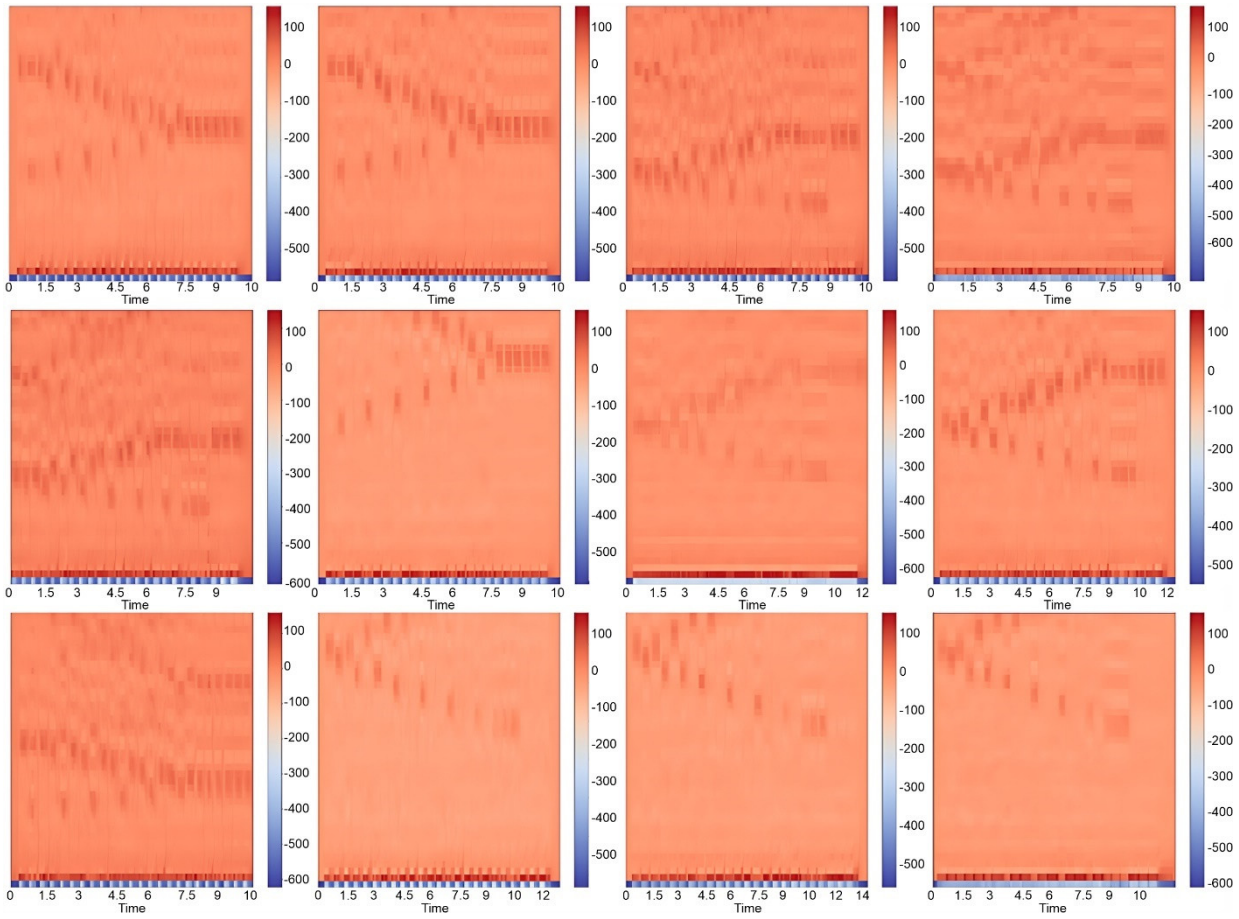

Figure 2. Extraction of sample features.

The image input has a size of 299 × 299 pixels, with three channels. In contrast to the preceding version (Inception v1 and v2), the Inception_v3 structure uses the convolutional kernel partitioning model to divide more significant integrals into small convolutions. For instance, a 3 × 3 line is divided into its constituent 1 × 3 and 3 × 1 lines. The segmentation model is utilized to minimize the parameter count. While increasing the speed of the training network, the spatial features are effectively retrieved. Simultaneously, Inception v3 improves the Inception structure with regional grid sizes of 8 × 8, 17 × 17, and 35 × 35.

3.4. ALO-Based Hyperparameter Tuning

In this study, the ALO algorithm is used to adjust the hyperparameters of the Inception v3 model. In this study, the ALO procedure is applied [25]. The lifecycle of an ant-lion includes two major phases: the larval stage and adulthood. In this work, the activities of ants as prey and the ant-lion as a hunter are modeled. When modeling such encounters, the ants will move close to the searching region, and the ant-lion will hunt them. The fitness of the ant-lion increases when it sets up further traps. Meanwhile, the ants move randomly to find food; an arbitrary move to model and motion is defined as follows:

$$x(t) = [0,\ totalsum(2r(t_1) - 1),\ \ldots,\ totalsum(2r(t_n) - 1)] \tag{1}$$

$$r(t) = \begin{cases} 1 & if\ rand > 0.5 \\ 0 & if\ rand \leq 0.5 \end{cases} \tag{2}$$

In this equation, the overall sum illustrates the overall density, n represents the maximum number of iterations, and t denotes the random motion phase. $r(t)$ shows the random function, and *rand* indicates an arbitrary value produced by uniform distribution between zero and one. The position of the ant is saved, and is later applied in the optimization:

$$M_{Ant} = \begin{bmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,d} \\ A_{2,1} & A_{2,2} & \dots & A_{2,d} \\ \vdots & \vdots & \vdots & \vdots \\ A_{n,1} & A_{n,2} & \dots & A_{n,d} \end{bmatrix} \tag{3}$$

$$M_{OA} = \begin{bmatrix} f([A_{1,1} & A_{1,2} & \dots & A_{1,d}]) \\ f([A_{2,1} & A_{2,2} & \dots & A_{2,d}]) \\ \vdots & \vdots & \vdots & \vdots \\ f([A_{n,1} & A_{n,2} & \dots & A_{n,d}]) \end{bmatrix} \tag{4}$$

In this equation, M_{Ant} denotes the matrix that comprises the position of every ant, $A_{i,j}$ indicates the value of the *j-th* variable, n specifies the ant count, and d indicates the variable count. The position and the fitness of the ant-lion are stored as shown below:

$$M_{Ant-lion} = \begin{bmatrix} ALi_{1,1} & ALi_{1,2} & \dots & ALi_{1,d} \\ ALi_{2,1} & ALi_{2,2} & \dots & ALi_{2,d} \\ \vdots & \vdots & \vdots & \vdots \\ ALi_{n,1} & ALi_{n,2} & \dots & ALi_{n,d} \end{bmatrix} \tag{5}$$

$$M_{OAL} = \begin{bmatrix} f([ALi_{1,1} & ALi_{1,2} & \dots & ALi_{1,d}]) \\ f([ALi_{2,1} & ALi_{2,2} & \dots & ALi_{2,d}]) \\ \vdots & \vdots & \vdots & \vdots \\ f([ALi_{n,1} & ALi_{n,2} & \dots & ALi_{n,d}]) \end{bmatrix} \tag{6}$$

where $M_{Ant-lion}$ denotes the matrix encompassing the position of every ant-lion, $ALi_{i,j}$ indicates the value of the *j-th* parameter, n characterizes the ant-lion count, and d represents the variable count. M_{OAL} denotes the matrix encompassing the fitness of every ant-lion, and f demonstrates the target function.

The ant updates their position by arbitrarily moving at every optimization stage. Since each searching space has a border, a normalizing process is employed for the random movement to keep them within the interval of the searching region. Here, the $Min - Max$ normalization technique is applied:

$$X_i^t = \frac{\left(X_i^t - a_i\right) * \left(d_i^t - c_i^t\right)}{\left(b_i^t - a_i\right)} + c_i \tag{7}$$

In Equation (7), b_i and a_i represent the maximum and the minimum arbitrary motion of the *j-th* parameter, respectively. Furthermore, d_i^t and c_i^t indicate the maximum and minimum of the *i-th* parameter in the *t-th* iteration, respectively. This formula should be applied iteratively to guarantee that arbitrary movement is preserved.

The ant-lion traps affect the arbitrary motion of the ants. To mathematically model this assumption, the subsequent equation is applied:

$$c_i^t = Ant - lion_j^t + c^t \; d_i^t = Ant - lion_j^t + d^t \tag{8}$$

In Equation (8), d^t and c^t represent the vector comprising the maximum and minimum value of each variable in the *t-th* iteration, respectively. Furthermore, d_i^t and c_i^t denote the maximum and the minimum value of each variable of the *j-th* ant, respectively. Furthermore, $Ant - lion_i^t$ indicates the position of the *j-th* chosen ant-lion at the *t-th* iteration. The equation shows that the ant moves close to the preferred ant-lion in a circular cloud described by the vectors *c* and *d*. For mathematical modeling, the circular cloud radius of arbitrary ant movement is adaptively decreased. The following equation is recommended:

$$c^t = \frac{c^t}{I} \, d^t = \frac{d^t}{I} \tag{9}$$

In Equation (9), *I* denotes a proportion, d^t and c^t represent the vector encompassing the maximum and minimum value of each variable at the *t-th* iteration, respectively. The last hunting phase occurs once an ant arrives underneath and is grabbed using its jaw. Then, the ant-lion pulls the ant into the soil and eats it. The ant-lion needs to update its position according to the final trap, in order to increase the probability of capturing new prey:

$$Ant - lion_j^t = Ant_i^t \; If \; f\left(Ant_i^t\right) > f\left(Ant - lion_j^t\right) \tag{10}$$

In Equation (10), *t* characterizes the current iteration, $Ant - lion_i^t$ indicates the location of the *j-th* preferred ant-lion at the *t-th* iteration, and Ant_i^t refers to the position of the *j-th* ant at the *t-th* iteration.

Here, the optimally chosen ant-lion in every iteration was stored and assumed to be elite. The ant-lion with the highest fitness values impacts the activity of each ant at that iteration. The following equation defines this motion:

$$Ant_i^t = \frac{R_A^t + R_E^t}{2} \tag{11}$$

In Equation (11), R_A^t describes the arbitrary motion near the chosen ant-lion according to the roulette wheel at the *t-th* iteration, R_E^t indicates the arbitrary motion close to the elite ant-lion at the *t-th* iteration, and Ant_i^t denotes the position of the *i-th* ant at the *t-th* iteration. The optimizer adds the nature of the ant-lion's behavior when hunting the target, given as follows: (1) arbitrary walk; (2) construction of trap; (3) trapping and in the trap; (4) catching the prey; and (5) rebuilding the trap.

The ALO approach derives a fitness function from accomplishing better classification performance. It describes a positive integer for characterizing the good performance of the candidate solution. In practical application, the reduction in the classification error rate can be considered as the fitness function. The best solution is that with the minimum error rate, while worse solutions possess higher error rates.

$$\begin{aligned} fitness(x_i) &= ClassifierErrorRate(x_i) \\ &= \frac{number \; of \; misclassified \; samples}{Total \; number \; of \; samples} * 100 \end{aligned} \tag{12}$$

3.5. LSTM-RNN-Based Verification Process

Finally, the LSTM-RNN model is exploited as a classifier for automated ASR. An input layer, two hidden layers, and an output layer make up the LSTM-RNN. Generally, a feedforward neural network (FFNN) can be defined as follows [26].

$$Y = F(X, \theta) \tag{13}$$

where $X = \{x_1, \, x_2, \, \cdots, \, x_n\}$ refers to an input set, $Y = \{y_1, \, y_2, \, \cdots, \, y_m\}$ stands for an output set, *F* stands for an FFNN module, and θ refers to a parameter set of the module. For a classification module, *Y* represents a set of classes. A convolutional neural network (CNN) is a kind of FFNN. It is employed for semantics segmentation, image classification,

and target recognition. The CNN method involves convolution and pooling layers, which sets it apart from other NNs. The convolution layers aim to extract the local features of the input dataset.

$$Y_F = Conv(X,\ \theta_{CONV}) \tag{14}$$

Conv is one convolution layer, Y_F is a feature subset extracted by the convolution layer from X. θ_{CONV} is a parameter set in the convolution layer. The pooling layer aims to compress the local feature, thus highlighting the feature.

$$Y_{CF} = Poo1(Y_{Fr}\theta_{Pool}) \tag{15}$$

Poo1 is one pooling layer. Y_{CF} represent a set of compressed features, and we present the fusion of CNN and pooling layer from Y_F. θ_{Pool} is a parameter set in the pooling layer. For a classification module, a CNN comprising *FC* and $Softmax$ layers is incorporated, and the front of the RNN forms a CRNN mechanism. $Y = F(X, \theta)$ is used to categorize the features. Here, the $Y = F(X,\ \theta)$ of CNN is denoted by:

$$Y = Softmax\left(FC\left(Pool\left(Conv(X, \theta_{CONV}), \theta_{pool}\right), \theta_{FC}\right)\right) \tag{16}$$

FC denotes one fully connected layer. SoftMax denotes one $Softmax$ layer. RNN is an alternative kind of FFNN. RNN is mainly employed on datasets with a sequential architecture, such as speech recognition, machine translation, etc. LSTM-RNN is a widely applied RNN method that is able to resolve the gradient vanishing problem with memory cells for storing long-term data. As a classification method, LSTM-RNN includes $Softmax$ and *FC* layers. The $y = F(X, \theta)$ of LSTM-RNN is given by:

$$y = Softmax(FC(LSTM(X, \theta_{LSTM}), \theta_{FC})) \tag{17}$$

Our approach uses a single-output long short-term memory recurrent neural network (LSTM-RNN) for decision making as shown in Figure 3. We used fixed-length inputs, while LSTMs are not constrained by this requirement. The utterances were converted into sets of features that acted as the input for the LSTM-RNN. A 25 ms frame was applied. The number of input coefficients is equal to the size of the LSTM-RNN input layer. Instead of a frame classifier, the network functions as a sequence classifier.

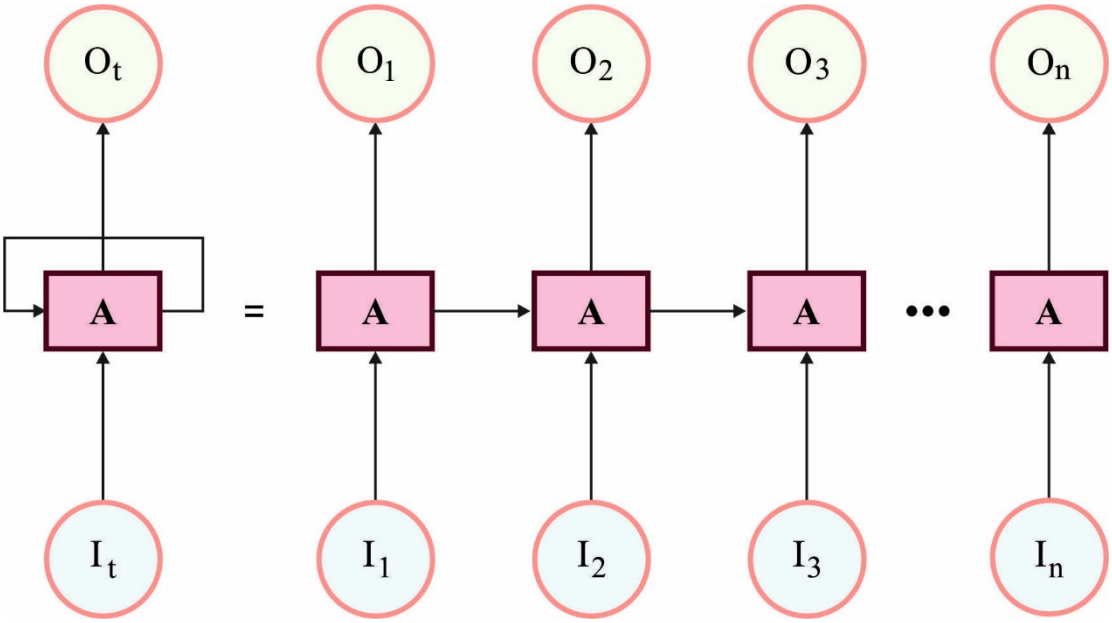

Figure 3. Framework of LSTM-RNN.

The decision-making process utilizes the loss function and the final hidden state of the LSTM model. The LSTM consists of two layers, each with 300 nodes. The SoftMax layer with the same number of classes is added. The memory blocks in the LSTM's hidden layers function as memory by preserving the network's present state. The SoftMax layer is then added afterwards, with the same number of classes. The SoftMax layer's output for a particular frame is the likelihood that the frame belongs to one of the speakers; however, this probability is dependent not only on the input frame, but also on every previous frame in the sequence. Every output can be determined by the system using both past and present inputs. When the system has access to the entire file, the results are calculated.

4. Results and Discussion

Evaluation Metrics

Utilizing four evaluation metrics—classification accuracy, error rate, precision, recall F-Score, G-measure, and execution time—we evaluated the effectiveness of our suggested LSTM-RNN approach. A True Positive (*TP*) is a result where the model correctly predicted the positive class. In contrast, a True Negative (*TN*) means the model correctly predicted the negative category. False Positives (*FP*s) occur when the model incorrectly forecasts the positive type. False Negatives (*FN*) result when the model mispredicts the negative variety. The definition of these measures is as follows:

Accuracy: The proportion of accurate predictions to all predictions is known as accuracy or classification accuracy. The accuracy is provided as a percentage in this study by

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \times 100\% \tag{18}$$

Precision: The precision standard is the ratio of correctly predicted positive observations to all positively anticipated statements.

$$Precision = \frac{TP}{TP + FP} \tag{19}$$

Recall: Recall is the proportion of times a positive instance is correctly distinguished from an actual positive example.

$$Recall = \frac{TP}{TP + FN} \tag{20}$$

F-Score: The harmonic mean between recall and precision is the F-Score. *F-Score* has a range of [0, 1]. It reveals the classifier's accuracy, how many cases it correctly classifies, its robustness, and if it needs to catch a sizable number of instances.

$$F - Score = \frac{2 * Precision * Recall}{Precision + Recal} \tag{21}$$

Error rate: The error rate is the percentage of incorrect responses in a process, system, or measurement. It is usually a percentage or ratio of errors to observations or attempts. Higher error rates may indicate inefficiencies in the process or system, while lower errors suggest greater accuracy and reliability. Statistics, engineering, computer science, and quality control employ error rates to evaluate and improve systems and processes.

G-Measure: G-Measure evaluates clustering methods by measuring their cluster purity and the number of clusters. The harmonic mean of accuracy and recall accounts for clustering's true positive and false positive rates. A G-Measure value of 1 implies perfect clustering, while 0 suggests random clustering, defined as follows:

$$\text{G-Measure} = \text{sqrt (precision} \times \text{recall)} \tag{22}$$

Precision is the ratio of true positives to the sum of true and false positives, and recall is the ratio of true positives to false negatives. The G-Measure is used alongside the silhouette score and Dunn index to evaluate clustering techniques.

Execution Time: In forensic investigations, an Execution Time-based Automated Forensic Speaker Verification Model analyses speech characteristics to verify a speaker's identity. Based on vocal patterns, the model can automatically identify a speaker. Execution time, the time it takes to execute a task, is used to quantify verification process efficiency of the model. This concept benefits forensic investigations when precision and quickness are crucial.

The speaker verification performance of the TTFEM-AFSV model was evaluated with five distinct speakers and a total of 400 samples, as illustrated in Table 1.

Table 1. Dataset details.

Label	No. of Speakers	No. of Samples
C-1	Speaker-1	80
C-2	Speaker-2	80
C-3	Speaker-3	80
C-4	Speaker-4	80
C-5	Speaker-5	80
Total Number of Samples		400

Figure 4 presents the confusion matrices formed by the TTFEM-AFSV model with different training (TR) and testing (TS) data sizes. The figure reports that the TTFEM-AFSV model demonstrated the best speaker recognition performance.

Table 2 shows the TTFEM-AFSV model's overall speaker recognition performance on 80% TR data and 20% TS data. The TTFEM-AFSV model's quick identification results on 80% TR data are presented in Figure 5. The findings suggest that the TTFEM-AFSV model was able to correctly recognize each class. For example, the TTFEM-AFSV model identified C-1 examples with $accu_y$ of 94.69%, ER of 5.31%, $prec_n$ of 88.06%, $reca_l$ of 86.76%, F_{score} of 87.41%, $G_{measure}$ of 87.41% and Exe_{time} of 2.143 ms. Meanwhile, the TTFEM-AFSV method identified C-2 samples with $accu_y$ of 96.25%, ER of 3.75%, $prec_n$ of 90.28%, $reca_l$ of 92.86%, F_{score} of 91.55%, $G_{measure}$ of 91.56% and Exe_{time} of 2.354 ms. Furthermore, the TTFEM-AFSV model identified C-3 samples with $accu_y$ of 95.31%, ER of 4.69%, $prec_n$ of 94.12%, $reca_l$ of 80.00%, F_{score} of 86.49%, $G_{measure}$ of 86.77% and Exe_{time} of 2.187 ms.

Table 2. Results analysis of the TTFEM-AFSV approach for different measures under 80:20 ratio of the TR/TS dataset.

Labels	Accuracy	Error Rate	Precision	Recall	F-Score	G-Measure
Training Phase (80%)						
C-1	94.69	05.31	88.06	86.76	87.41	87.41
C-2	96.25	03.75	90.28	92.86	91.55	91.56
C-3	95.31	04.69	94.12	80.00	86.49	86.77
C-4	95.31	04.69	84.13	91.38	87.60	87.68
C-5	94.06	05.94	83.58	87.50	85.50	85.52
Average	95.12	04.87	88.03	87.70	87.71	87.79
Testing Phase (20%)						
C-1	98.75	01.25	92.31	100.00	96.00	96.08

Table 2. *Cont.*

Labels	Accuracy	Error Rate	Precision	Recall	F-Score	G-Measure
C-2	100.00	00.00	100.00	100.00	100.00	100.00
C-3	100.00	00.00	100.00	100.00	100.00	100.00
C-4	100.00	00.00	100.00	100.00	100.00	100.00
C-5	98.75	01.25	100.00	93.75	96.77	96.82
Average	99.50	00.50	98.46	98.75	98.55	98.58

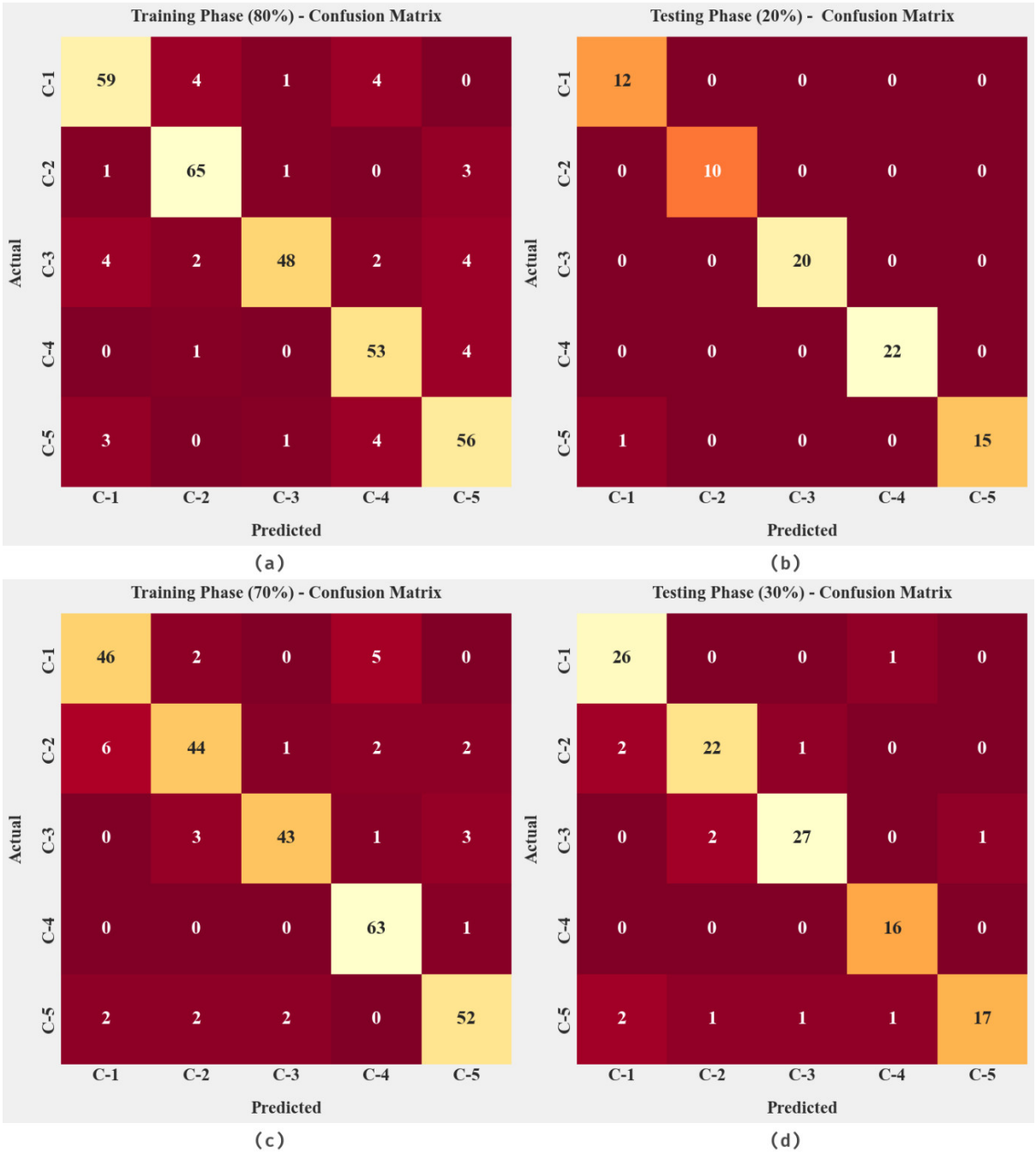

Figure 4. Confusion matrices of the TTFEM-AFSV approach: (**a**) 80% TR data, (**b**) 20% TS data, (**c**) 70% TR data, and (**d**) 30% TS data.

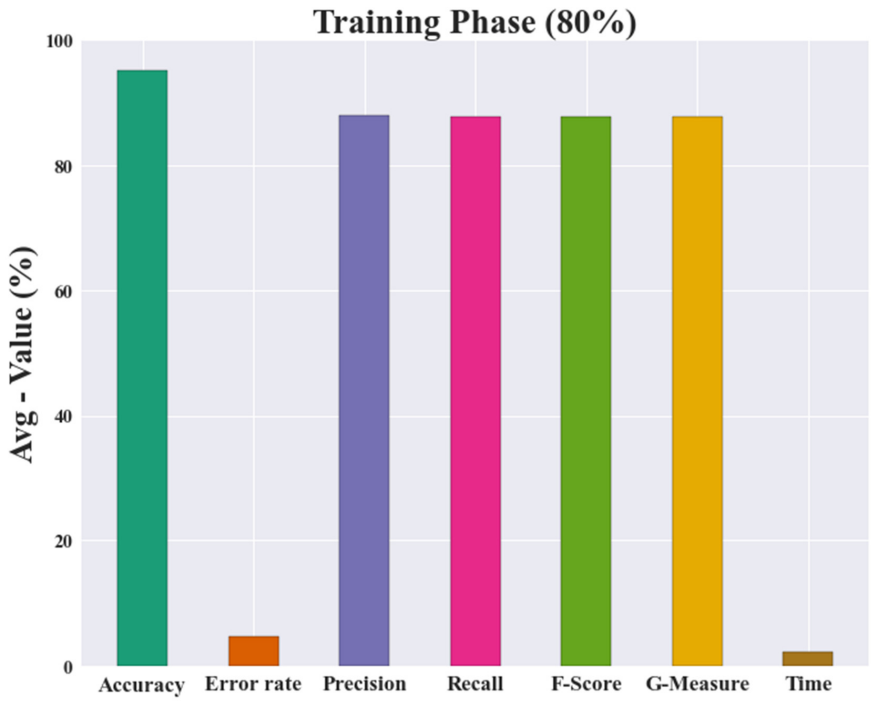

Figure 5. Average analysis of TTFEM-AFSV approach under 80% TR data.

The TTFEM-AFSV model on 20% of the TS dataset produced rapid recognition results, as shown in Figure 6. It can be inferred from the results that the TTFEM-AFSV approach was able to identify all classes proficiently. For instance, the TTFEM-AFSV method identified C-1 samples with $accu_y$ of 98.75%, ER of 1.25%, $prec_n$ of 92.31%, $reca_l$ of 100.00%, F_{score} of 96.00%, $G_{measure}$ of 96.08% and Exe_{time} of 3.107 ms. Meanwhile, the TTFEM-AFSV method identified C-2 samples with $accu_y$ of 100%, ER of 0%, $prec_n$ of 100%, $reca_l$ of 100%, F_{score} of 100%, $G_{measure}$ of 100% and Exe_{time} of 3.208 ms. Furthermore, the TTFEM-AFSV method identified C-3 samples with $accu_y$ of 100%, ER of 0%, $prec_n$ of 100%, $reca_l$ of 100%, F_{score} of 100%, $G_{measure}$ of 100% and Exe_{time} of 3.128 ms.

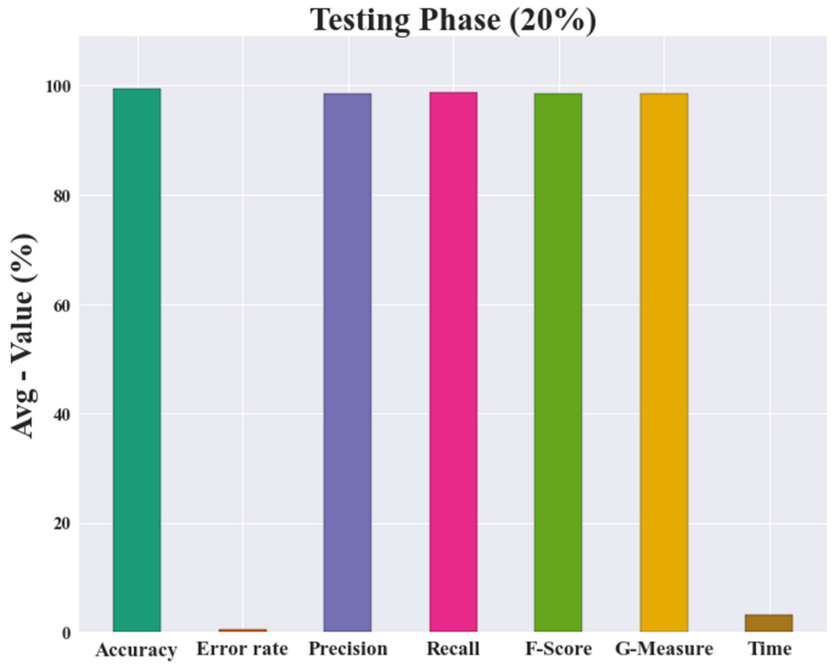

Figure 6. Average analysis of TTFEM-AFSV approach under 20% TS data.

The training accuracy (TA) and validation accuracy (VA) achieved by the TTFEM-AFSV approach under 80:20 TR/TS data is presented in Figure 7. The experimental outcomes imply that the TTFEM-AFSV technique achieved the maximum values of TA and VA. Significantly, the VA seemed to be greater than TA.

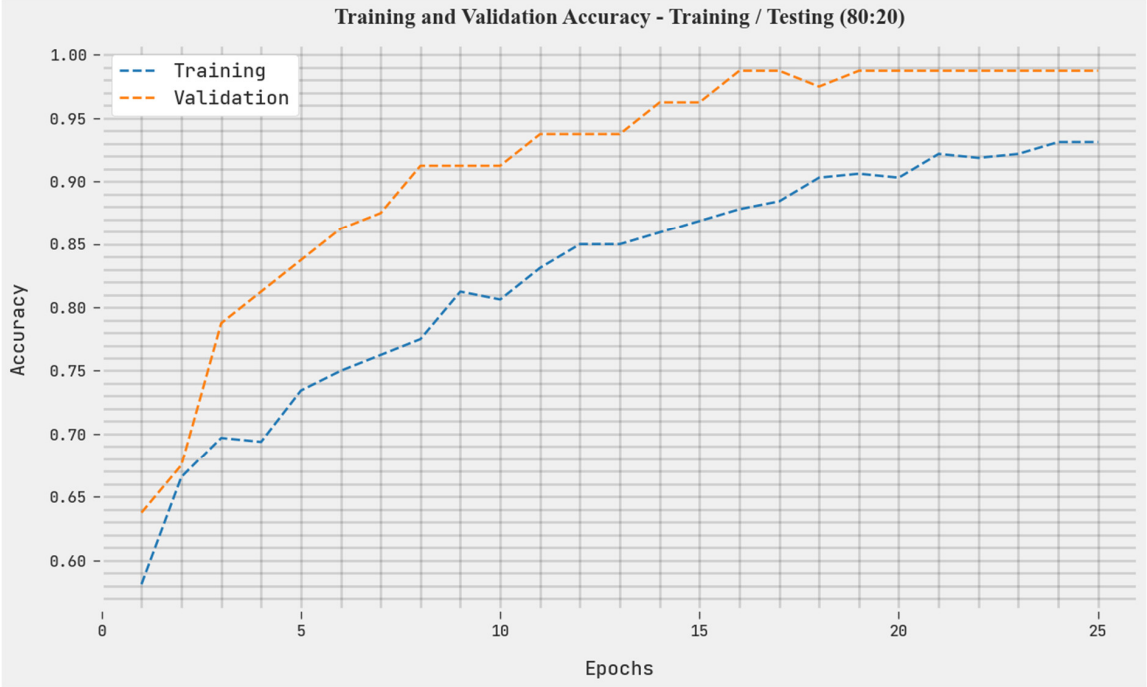

Figure 7. TA and VA analysis of TTFEM-AFSV approach under 80:20 TR/TS data.

The training loss (TL) and validation loss (VL) obtained by the TTFEM-AFSV method under 80:20 TR/TS data are presented in Figure 8. The experimental outcomes imply that the TTFEM-AFSV algorithm achieved the minimum values of TL and VL. Remarkably, the VL is less than the TL.

Figure 8. TL and VL analysis of TTFEM-AFSV approach under 80:20 TR/TS data.

A clear precision–recall examination of the TTFEM-AFSV technique under 80:20 TR/TS data is portrayed in Figure 9. The figure indicates that the TTFEM-AFSV process resulted in improved values of precision–recall values for all classes.

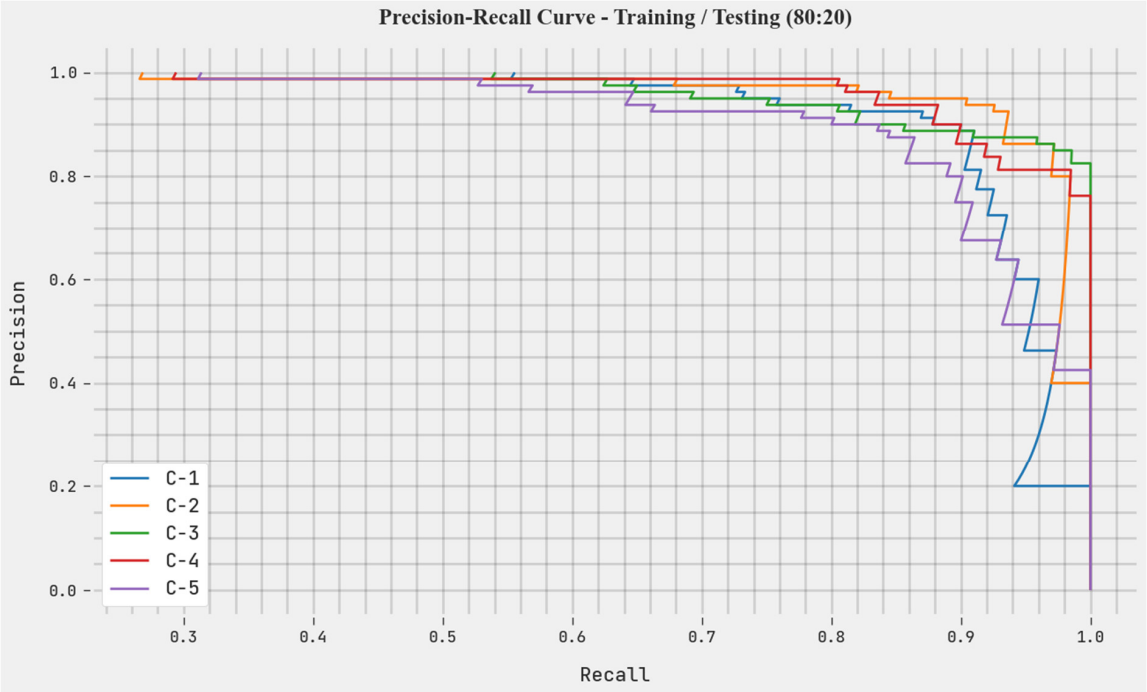

Figure 9. Precision–recall analysis of the TTFEM-AFSV approach under 80:20 TR/TS data.

A brief ROC analysis of the TTFEM-AFSV approach under 80:20 TR/TS data is presented in Figure 10. The outcomes indicate that the TTFEM-AFSV method demonstrated good ability to classify distinct classes.

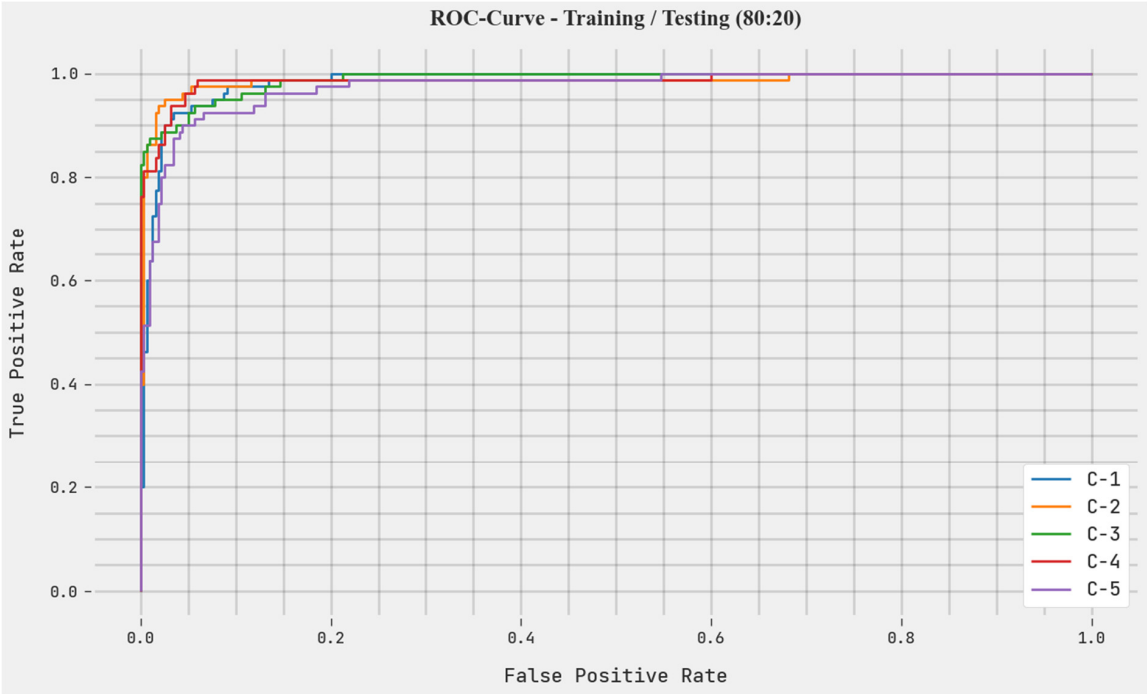

Figure 10. ROC analysis of the TTFEM-AFSV approach under 80:20 TR/TS data.

The overall speaker recognition performance of the TTFEM-AFSV method on 70% TR data and 30% TS data is presented in Table 3. Figure 11 indicates the rapid recognition outcomes of the TTFEM-AFSV approach on 70% of the TR dataset. The results illustrate that the TTFEM-AFSV method was able to proficiently identify all classes. For example, the TTFEM-AFSV technique identified C-1 samples with $accu_y$ of 94.64%, ER of 5.36%, $prec_n$ of 85.19%, $reca_l$ of 86.79%, F_{score} of 85.98%, and $G_{measure}$ of 85.99%. Meanwhile, the TTFEM-AFSV method identified C-2 samples with $accu_y$ of 93.57%, ER of 6.43%, $prec_n$ of 86.27%, $reca_l$ of 80.00%, F_{score} of 83.02%, and $G_{measure}$ of 83.08%. Furthermore, the TTFEM-AFSV method identified C-3 samples with $accu_y$ of 96.43%, ER of 3.57%, $prec_n$ of 93.48%, $reca_l$ of 86.00%, F_{score} of 89.58%, and $G_{measure}$ of 89.66%.

Table 3. Results analysis of the TTFEM-AFSV approach with different measures under 70:30 TR/TS dataset.

Labels	Accuracy	Error Rate	Precision	Recall	F-Score	G-Measure
Training Phase (70%)						
C-1	94.64	05.36	85.19	86.79	85.98	85.99
C-2	93.57	06.43	86.27	80.00	83.02	83.08
C-3	96.43	03.57	93.48	86.00	89.58	89.66
C-4	96.79	03.21	88.73	98.44	93.33	93.46
C-5	95.71	04.29	89.66	89.66	89.66	89.66
Average	95.43	04.57	88.67	88.18	88.31	88.37
Training Phase (30%)						
C-1	95.83	04.17	86.67	96.30	91.23	91.35
C-2	95.00	05.00	88.00	88.00	88.00	88.00
C-3	95.83	04.17	93.10	90.00	91.53	91.54
C-4	98.33	01.67	88.89	100.00	94.12	94.28
C-5	95.00	05.00	94.44	77.27	85.00	85.43
Average	96.00	04.00	90.22	90.31	89.97	90.12

Figure 12 presents the rapid recognition results obtained using the TTFEM-AFSV system on 30% of the TS dataset. The results indicate that the TTFEM-AFSV technique was able to identify all classes proficiently. For example, the TTFEM-AFSV method identified C-1 samples with $accu_y$ of 95.83%, ER of 4.17%, $prec_n$ of 86.67%, $reca_l$ of 96.30%, F_{score} of 91.23%, and $G_{measure}$ of 91.35%. Meanwhile, the TTFEM-AFSV method identified C-2 samples with $accu_y$ of 100%, ER of 5.00%, $prec_n$ of 95.00%, $reca_l$ of 88.00%, F_{score} of 88.00%, and $G_{measure}$ of 88.00%. Furthermore, the TTFEM-AFSV method identified C-3 samples with $accu_y$ of 95.83%, ER of 4.17%, $prec_n$ of 93.10%, $reca_l$ of 90.00%, F_{score} of 91.53%, and $G_{measure}$ of 91.54%.

The TA and VA achieved by the TTFEM-AFSV approach under a 70:30 TR/TS data are presented in Figure 13. The experimental outcomes indicated that the TTFEM-AFSV technique achieved the maximum values of TA and VA. Specifically, the VA seemed to be greater than the TA.

The training loss (TL) and validation loss (VL) obtained when using the TTFEM-AFSV method under a 70:30 TR/TS data are illustrated in Figure 14. The experimental outcomes indicated that the TTFEM-AFSV algorithm achieved the lowest values of TL and VL. Notably, the VL was less than TL.

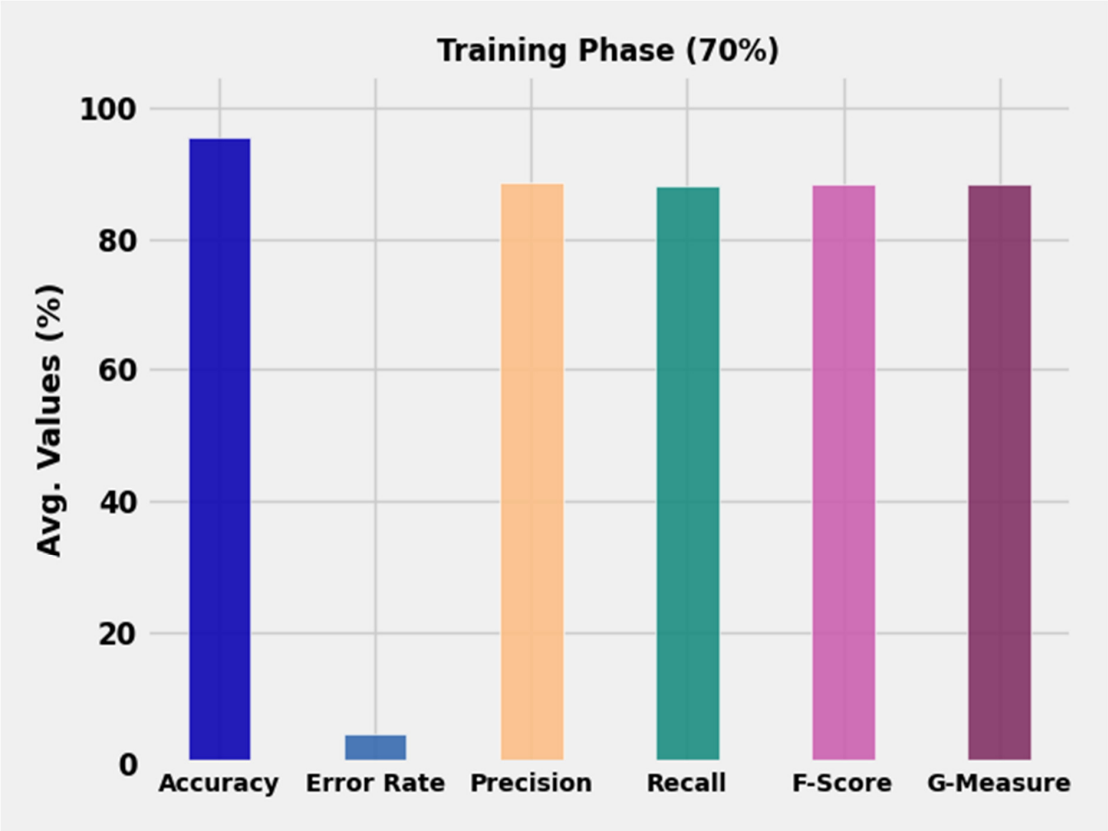

Figure 11. Average analysis of the TTFEM-AFSV approach under 70% TR data.

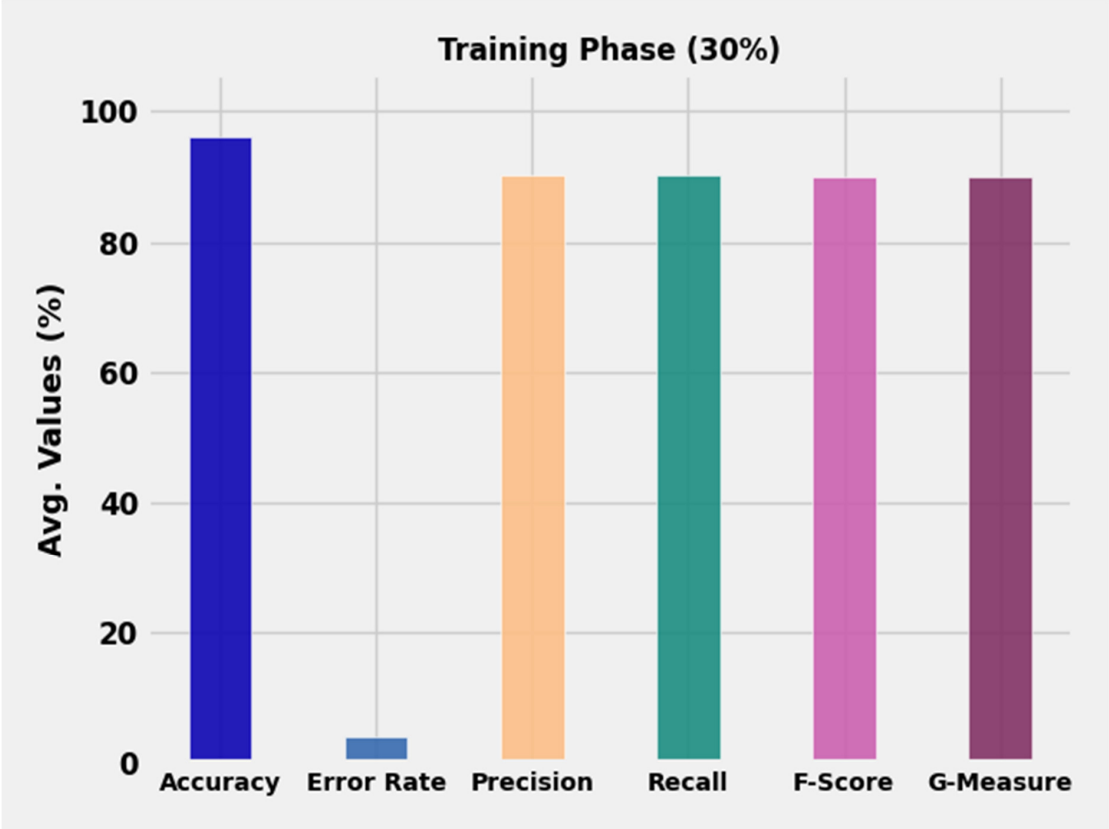

Figure 12. Average analysis of the TTFEM-AFSV approach under 30% TS data.

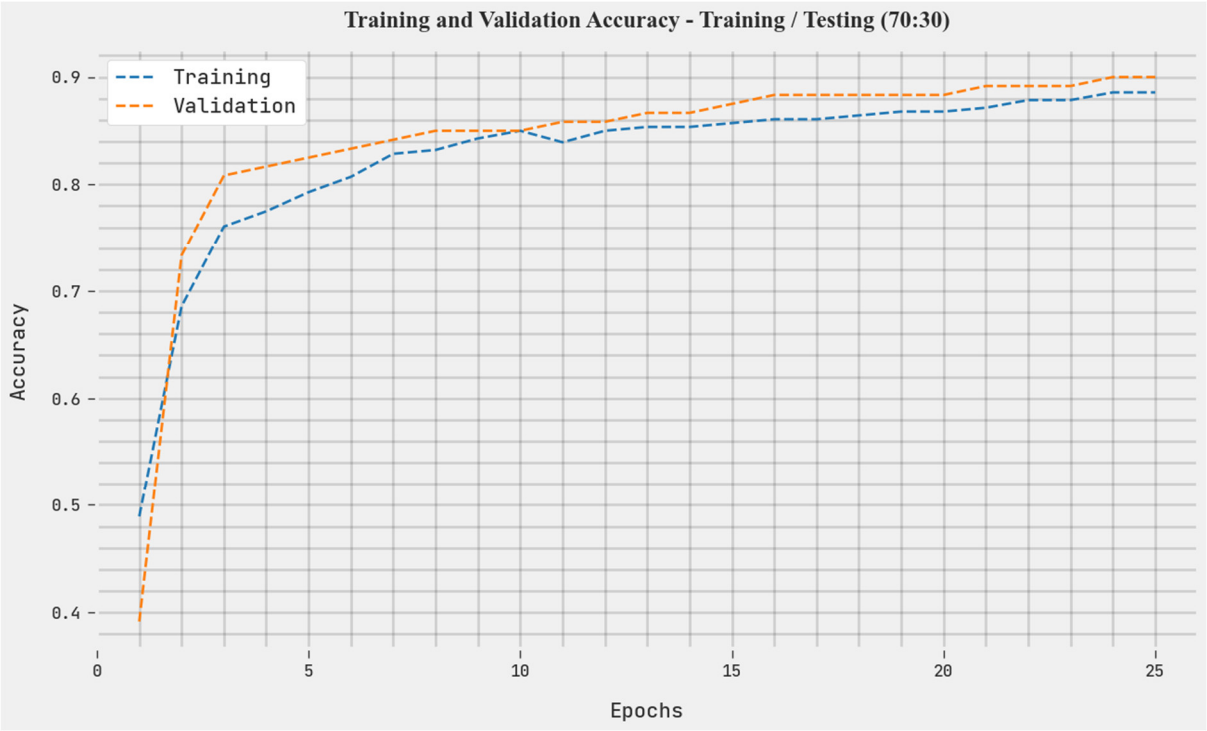

Figure 13. TA and VA analysis of TTFEM-AFSV approach under 70:30 TR/TS data.

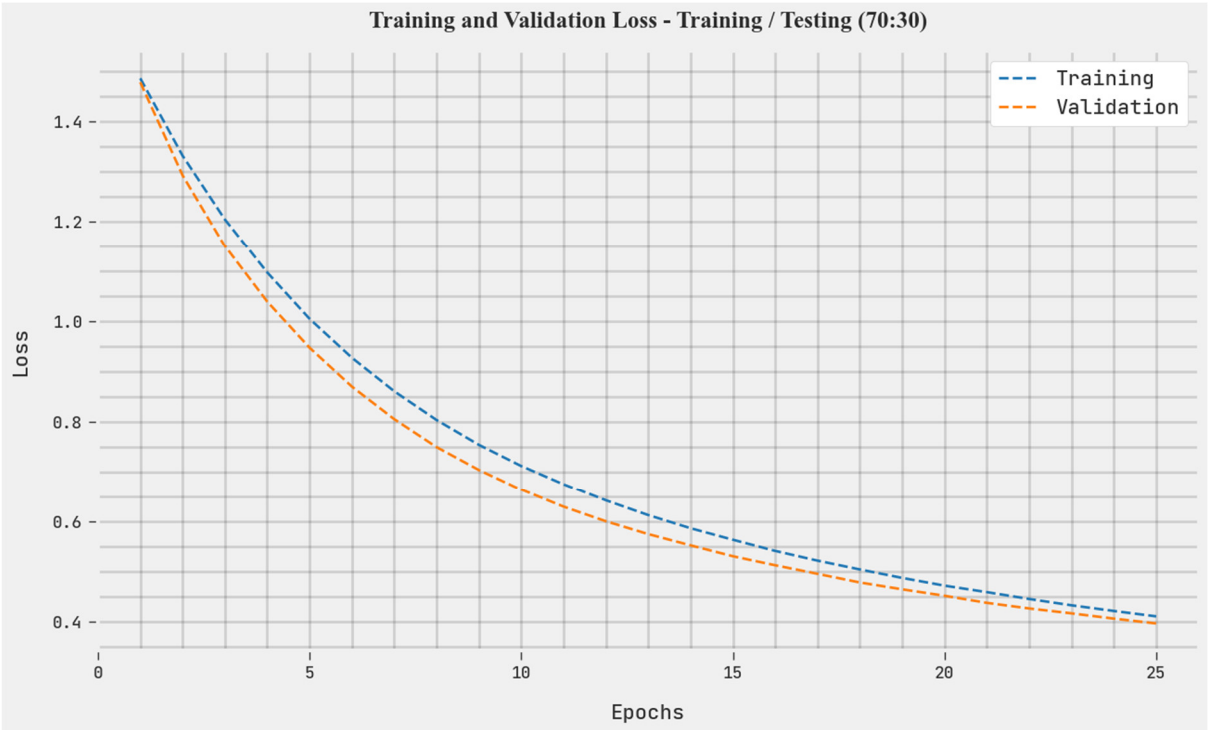

Figure 14. TL and VL analysis of the TTFEM-AFSV approach under 70:30 TR/TS data.

A clear precision–recall examination of the TTFEM-AFSV technique under 70:30 TR/TS data is presented in Figure 15. The figure indicates that the TTFEM-AFSV system demonstrated enhanced precision–recall values under all classes.

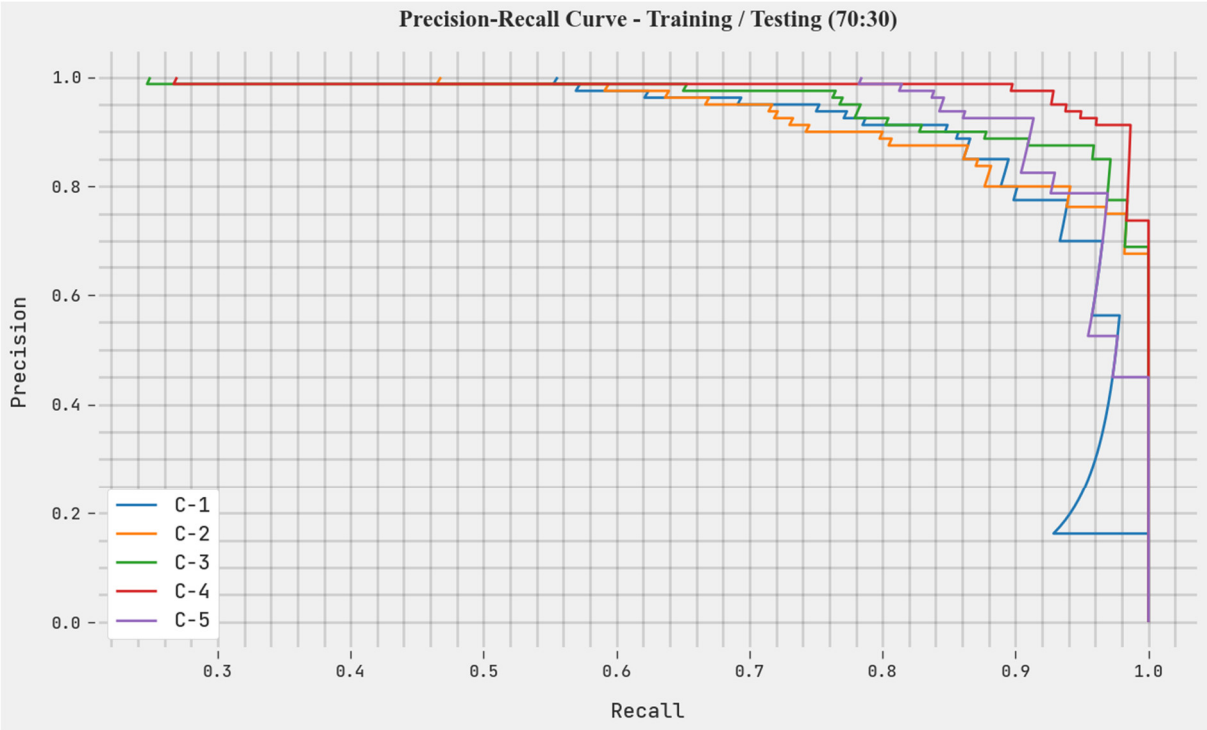

Figure 15. Precision–recall analysis of the TTFEM-AFSV approach under 70:30 TR/TS data.

A brief ROC analysis of the TTFEM-AFSV approach under 70:30 TR/TS data is presented in Figure 16. The results indicate the ability of the TTFEM-AFSV method to classify dissimilar classes.

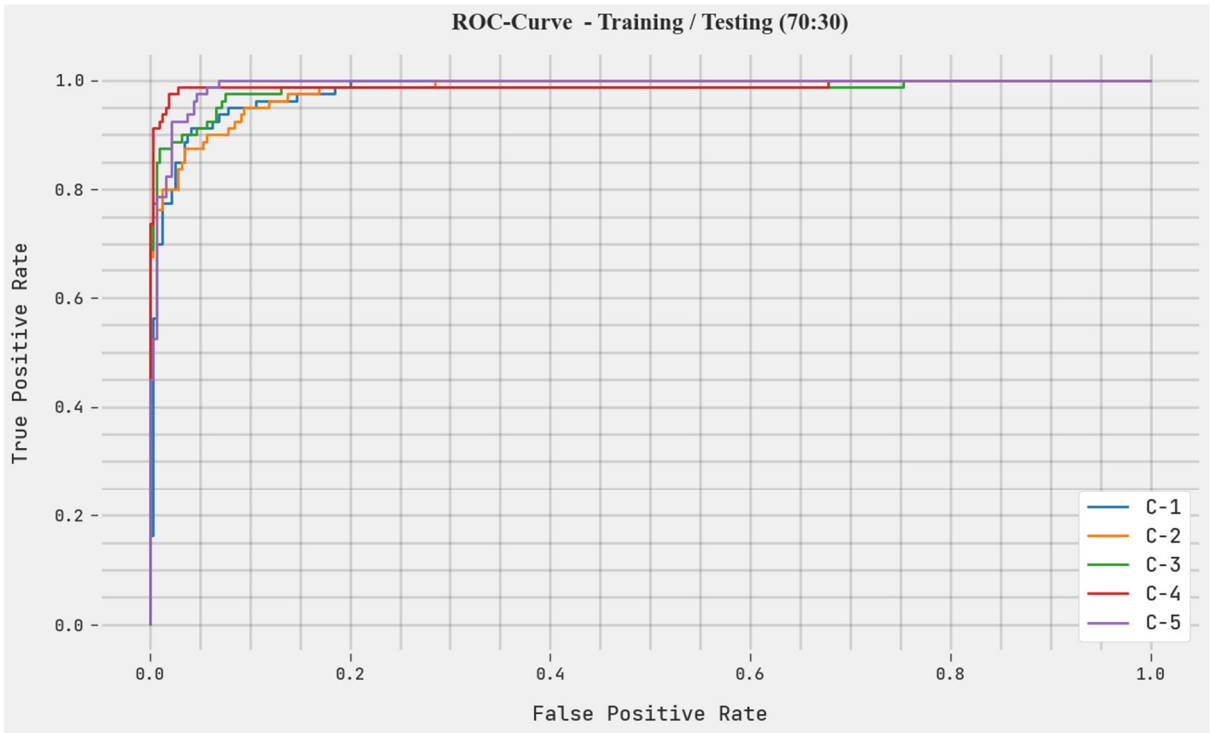

Figure 16. ROC analysis of the TTFEM-AFSV approach under 70:30 TR/TS data.

To illustrate the TTFEM-AFSV model's improved performance, a comparative study was performed, the results of which are shown in Table 4 [11,26]. Figure 17 presents a comparative $accu_y$ analysis of the TTFEM-AFSV model with other recent models. The figure indicates that the MFCC-SOFM-MLP-GD model had the lowest $accu_y$ of 96.92%. At the same time, the MFCC-SOFM-MLP-GDM, MFCC-SOFM-MLP-BR, MFCC-FW, and fusion models obtained slightly improved $accu_y$ values of 97.05%, 97.62%, 97.32%, and 97.81%, respectively. Although the DWT-MFCC technique presented a reasonable $accu_y$ of 98.87%, the TTFEM-AFSV model showed superior results, with a higher $accu_y$ of 99.50%.

Table 4. Comparison of the accuracy analysis of the TTFEM-AFSV approach with recent methodologies.

Methods	Accuracy	Error Rate
TTFEM-AFSV	99.50	0.50
MFCC-SOFM-MLP-GD	96.92	3.08
MFCC-SOFM-MLP-GDM	97.05	2.95
MFCC-SOFM-MLP-BR	97.62	2.38
MFCC-FW	97.32	2.68
DWT-MFCC	98.87	1.13
FUSION	97.81	2.19

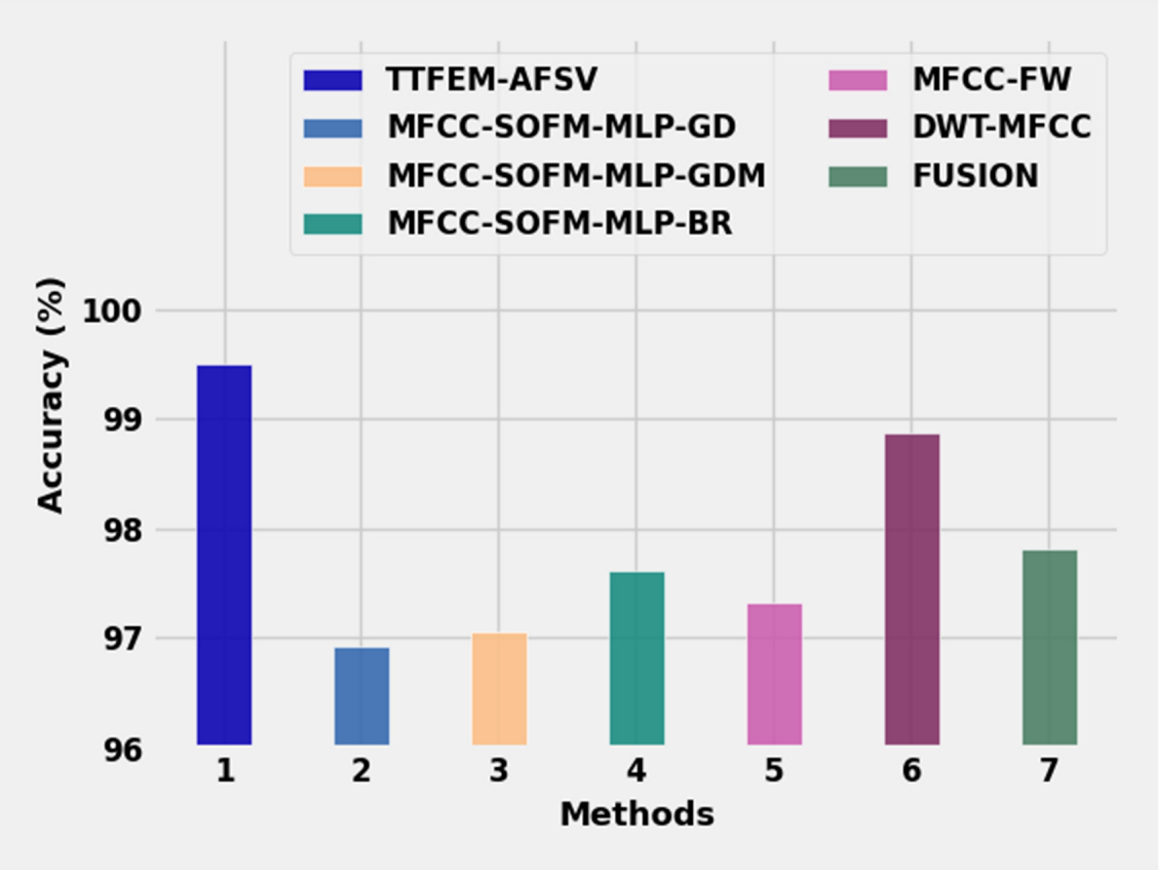

Figure 17. Comparison of the $Accu_y$ analysis of the TTFEM-AFSV approach with other recent methodologies.

Figure 18 depicts a comparison of the error rate (ER) of the TTFEM-AFSV method with that of other current models. The figure shows that the MFCC-SOFM-MLP-GD method had a greater ER of 3.08%. At the same time, the MFCC-SOFM-MLP-GDM, MFCC-SOFM-MLP-BR, MFCC-FW, and fusion approaches attained slightly reduced ER values of 2.95%, 2.38%, 2.68%, and 2.19%, respectively. Although the DWT-MFCC method demonstrated a

reasonable ER of 1.13%, the TTFEM-AFSV system presented more remarkable outcomes, with the lowest ER of 0.50%. Therefore, the TTFEM-AFSV model showed more effective speaker verification performance compared to the other models.

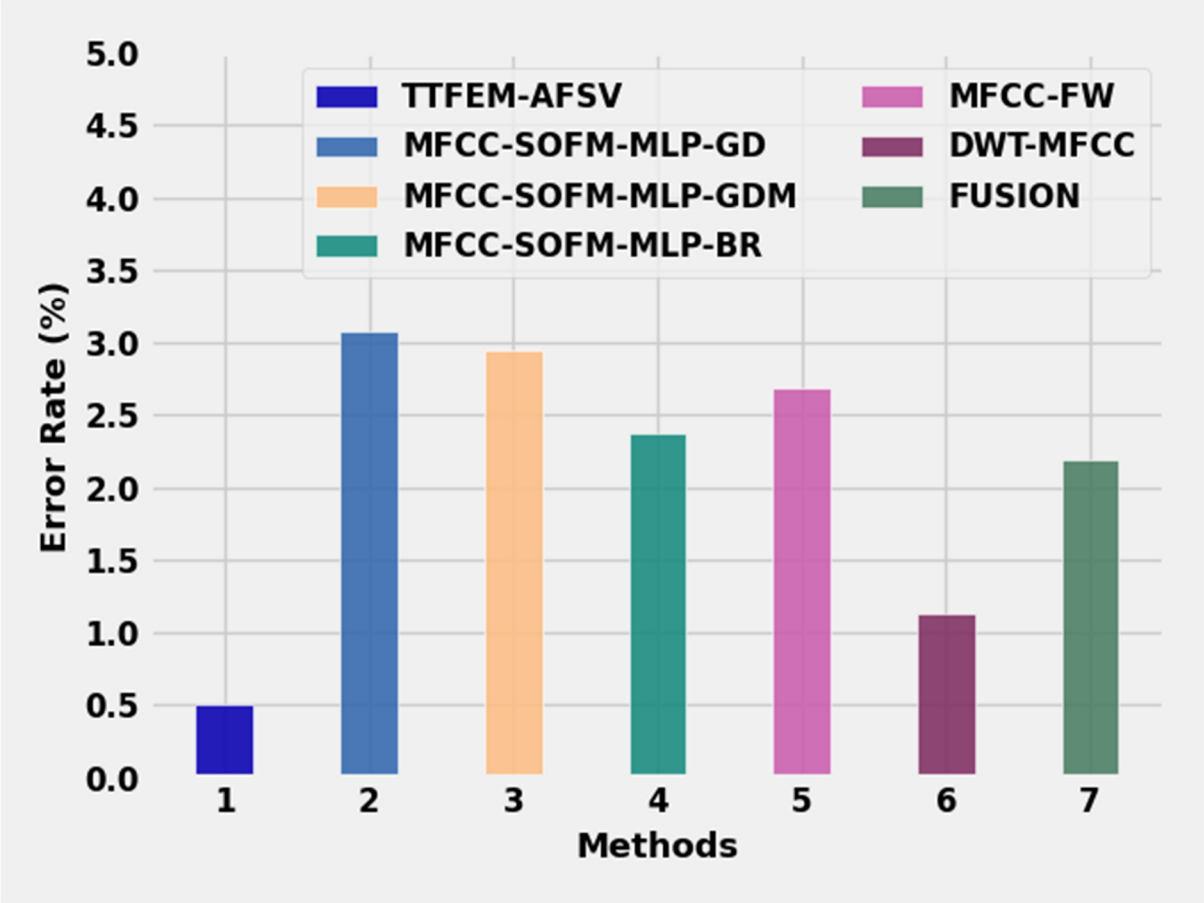

Figure 18. Comparison of the error rate of the TTFEM-AFSV approach with recent methodologies.

The proposed TTFEM-AFSV methodology is compared to other current approaches using RMSE, as shown in Figure 19 and Table 5. The figure illustrates how the AMF Filter technique improved performance with less RMSE noise. For instance, the proposed TTFEM-AFSV model's RMSE value at a noise level of 10% was 31.543%, while the MFCC-SOFM-MLP-GD, MFCC-SOFM-MLP-GDM, MFCC-SOFM-MLP-BR, and MFCC-FW models all achieved slightly improved RMSE values of 81.342%, 58.425%, 66.526%, and 51.324%, respectively. On the other hand, the proposed TTFEM-AFSV model demonstrated the best performance for various sounds with low RMSE values. Similarly, with noise of 90%, the RMSE value of the proposed TTFEM-AFSV was 23.655%, whereas, for the MFCC-SOFM-MLP-GD, MFCC-SOFM-MLP-GDM, MFCC-SOFM-MLP-BR, and MFCC-FW models, it was 71.8727%, 52.633%, 61.938%, and 45.746%, respectively.

The results of the accuracy study for the TTFEM-AFSV based on AMF filtering are shown in Figure 20 and Table 6. The accuracy level for Speaker1 was 76.24% for unfiltered data without AMF filtering and 93.42% for unfiltered data with AMF filtering, which is 17.18% higher than unfiltered data without AMF filtering. The accuracy level for Speaker 3 was 76.19% for unfiltered data without AMF filtering and 93.59% for unfiltered data with AMF filtering, which is 17.14% higher than unfiltered data without AMF filtering. The accuracy level for Speaker5 was 77.02% for unfiltered data without AMF filtering and 94.36% for unfiltered data with AMF filtering, which is 17.34% higher than the unprocessed data without AMF filtering.

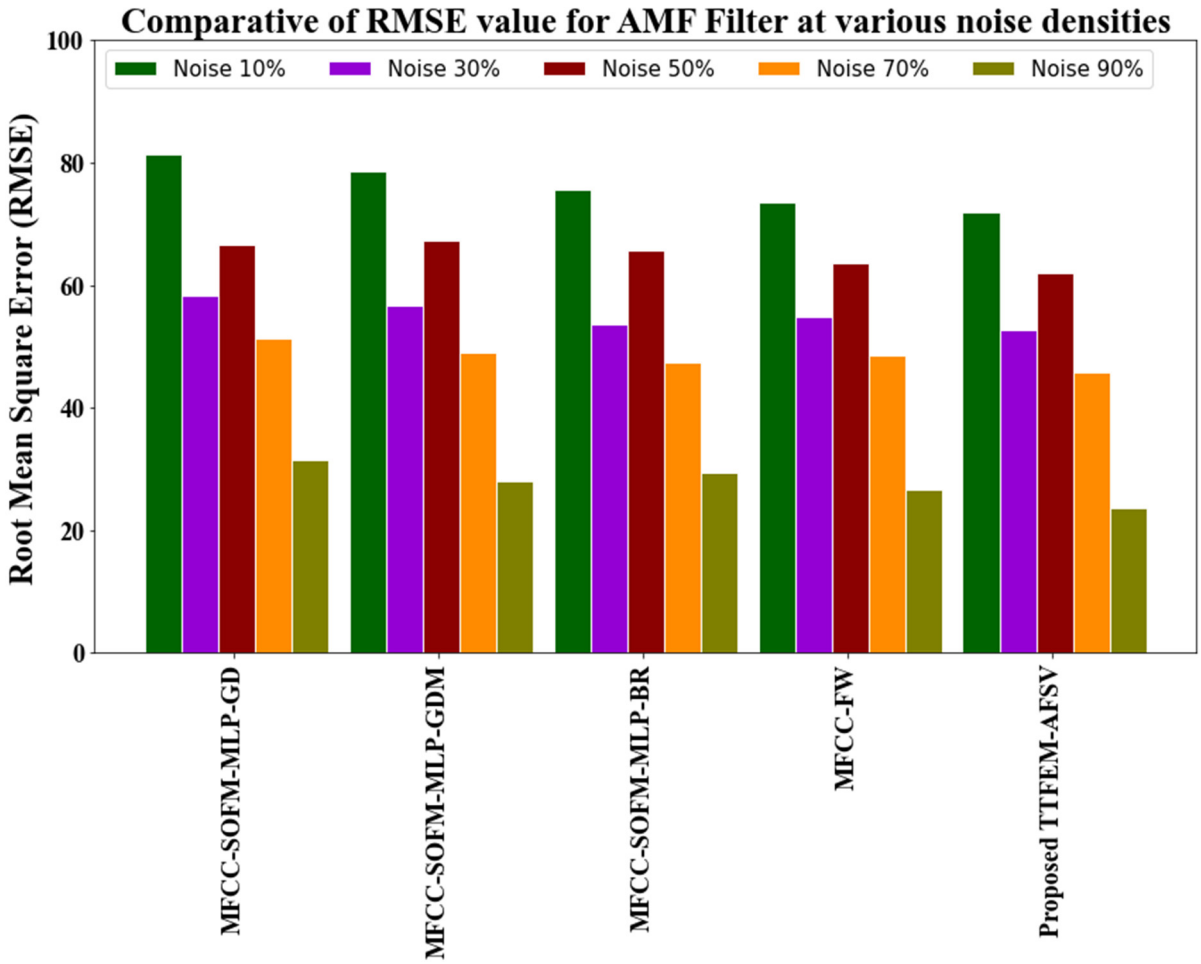

Figure 19. Comparison of RMSE values for AMF filtering at various noise densities.

Table 5. Comparison of RMSE values for AMF filter at various noise densities.

Methods	Noise 10%	Noise 30%	Noise 50%	Noise 70%	Noise 90%
MFCC-SOFM-MLP-GD	81.342	78.543	75.536	73.625	71.8727
MFCC-SOFM-MLP-GDM	58.425	56.837	53.625	54.938	52.633
MFCC-SOFM-MLP-BR	66.526	67.425	65.827	63.672	61.938
MFCC-FW	51.324	48.928	47.423	48.533	45.746
TTFEM-AFSV	31.543	27.983	29.326	26.543	23.655

Table 6. Accuracy analysis based on AMF filtering.

No of Speakers	Unfiltered Data without AMF Filtering	Unfiltered Data with AMF Filtering
Speaker 1	76.24	93.42
Speaker 2	76.32	93.61
Speaker 3	76.19	93.59
Speaker 4	76.58	94.21
Speaker 5	77.02	94.36

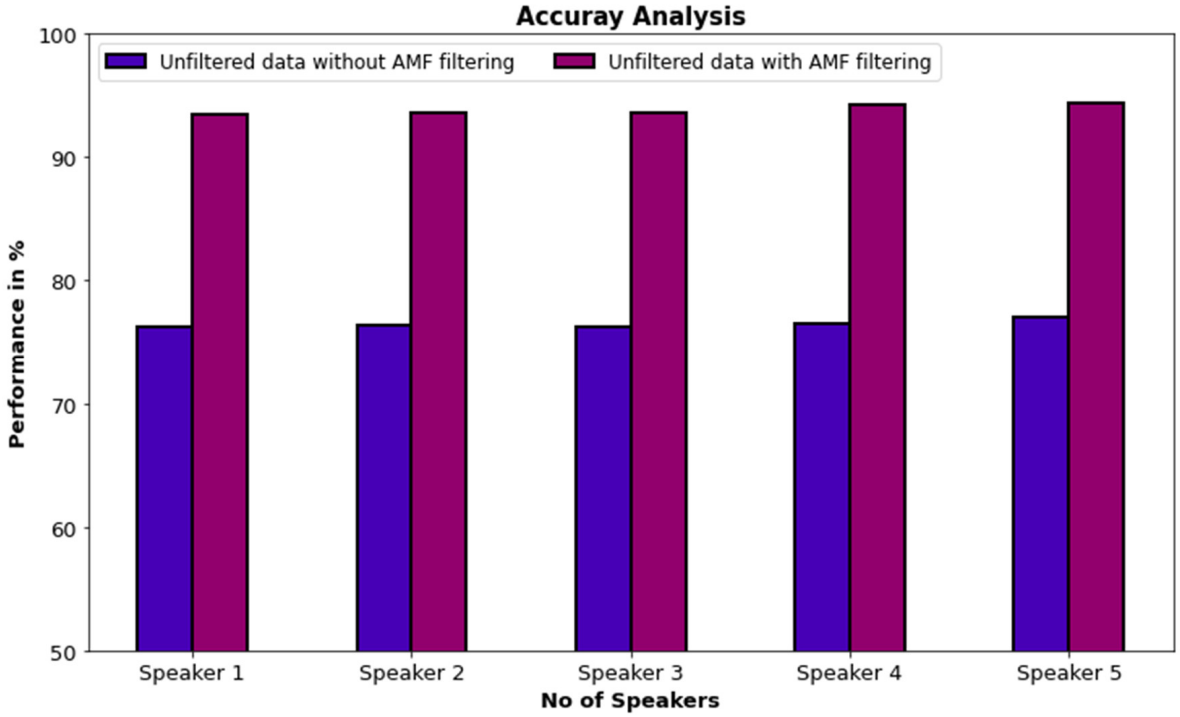

Figure 20. Accuracy analysis based on AMF filtering.

5. Conclusions

In this article, a new TTFEM-AFSV model was developed to verify speakers for forensic applications. The TTFEM-AFSV model initially employs the AMF technique to discard the noise present in speech signals. Next, the MFCC and spectrograms are considered as the input to the Inception v3 model. Then, the ALO algorithm is utilized to fine-tune the hyperparameters related to the Inception v3 model. Finally, the LSTM-RNN model is exploited as a classifier to perform automated ASR. The performance validation of the TTFEM-AFSV model was carried out by means of a series of experiments. The comparison study revealed the improved performance of the TTFEM-AFSV model compared to other recent approaches. Thus, the presented TTFEM-AFSV model can be exploited for effective ASR in real-time forensic applications. In the future, an ensemble of fusion-based DL models could be derived to boost the overall performance of the TTFEM-AFSV model.

Author Contributions: Conceptualization, G. and S.B.; methodology, G. and S.B.; software, G.; validation, G., S.B. and R.A.; formal analysis, G., S.B. R.A. writing—original draft preparation, G., S.B; writing—review and editing, S.B. and R.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data available on request due to restrictions e.g., privacy.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Machado, T.J.; Vieira Filho, J.; de Oliveira, M.A. Forensic speaker verification using ordinary least squares. *Sensors* **2019**, *19*, 4385. [CrossRef]
2. Wang, Z.; Xia, W.; Hansen, J.H. Cross-domain adaptation with discrepancy minimization for text-independent forensic speaker verification. *arXiv* **2020**, arXiv:2009.02444.
3. Stefanus, I.; Sarwono, R.J.; Mandasari, M.I. GMM-based automatic speaker verification system development for forensics in Bahasa Indonesia. In Proceedings of the 2017 5th International Conference on Instrumentation, Control, and Automation (ICA), Yogyakarta, Indonesia, 9–11 August 2017; pp. 56–61.

4. Algabri, M.; Mathkour, H.; Bencherif, M.A.; Alsulaiman, M.; Mekhtiche, M.A. Automatic speaker recognition for mobile forensic applications. *Mob. Inf. Syst.* **2017**, *2017*, 6986391. [CrossRef]
5. Gaurav, B.S.; Agarwal, R. An efficient speaker identification framework based on Mask R-CNN classifier parameter optimized using hosted cuckoo optimization (HCO). *J. Ambient Intell. Human. Comput.* **2022**, *13*, 1–13. [CrossRef]
6. Susanto, S.; Wang, Z.; Wang, Y.; Nanda, D.S. Forensic Linguistic Inquiry into the Validity of F0 as Discriminatory Potential in the System of Forensic Speaker Verification. *J. Forensic Sci. Crim. Investig.* **2017**, *5*, 555664.
7. Nagrani, A.; Chung, J.S.; Xie, W.; Zisserman, A. Voxceleb: Large-scale speaker verification in the wild. *Comput. Speech Lang.* **2020**, *60*, 101027. [CrossRef]
8. Athulya, M.S.; Sathidevi, P.S. Speaker verification from codec distorted speech for forensic investigation through serial combination of classifiers. *Digit. Investig.* **2018**, *25*, 70–77. [CrossRef]
9. Hautamäki, R.G.; Sahidullah, M.; Hautamäki, V.; Kinnunen, T. Acoustical and perceptual study of voice disguise by age modification in speaker verification. *Speech Commun.* **2017**, *95*, 1–15. [CrossRef]
10. Das, R.K.; Prasanna, S.M. Speaker verification from short utterance perspective: A review. *IETE Tech. Rev.* **2018**, *35*, 599–617. [CrossRef]
11. Susanto, S.; Nanda, D.S. December. Analysing Forensic Speaker Verification by Utilizing Artificial Neural Network. In *International Congress of Indonesian Linguistics Society (KIMLI 2021)*; Atlantis Press: Amsterdam, The Netherlands, 2021; pp. 128–132.
12. Al-Ali, A.K.H.; Dean, D.; Senadji, B.; Chandran, V.; Naik, G.R. Enhanced forensic speaker verification using a combination of DWT and MFCC feature warping in the presence of noise and reverberation conditions. *IEEE Access* **2017**, *5*, 15400–15413. [CrossRef]
13. Huang, S.; Dang, H.; Jiang, R.; Hao, Y.; Xue, C.; Gu, W. Multilayer Hybrid Fuzzy Classification Based on SVM and Improved PSO for Speech Emotion Recognition. *Electronics* **2021**, *10*, 2891. [CrossRef]
14. Swain, M.; Maji, B.; Kabisatpathy, P.; Routray, A. A DCRNN-based ensemble classifier for speech emotion recognition in Odia language. *Complex Intell. Syst.* **2022**, *8*, 4237–4249. [CrossRef]
15. Mardhotillah, R.; Dirgantoro, B.; Setianingsih, C. Speaker Recognition for Digital Forensic Audio Analysis using Support Vector Machine. In Proceedings of the 2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Yogyakarta, Indonesia, 10–11 December 2020; pp. 514–519.
16. Saleem, S.; Subhan, F.; Naseer, N.; Bais, A.; Imtiaz, A. Forensic speaker recognition: A new method based on extracting accent and language information from short utterances. *Forensic Sci. Int. Digit. Investig.* **2020**, *34*, 300982. [CrossRef]
17. Khan, F.; Tarimer, I.; Alwageed, H.S.; Karadağ, B.C.; Fayaz, M.; Abdusalomov, A.B.; Cho, Y.-I. Effect of Feature Selection on the Accuracy of Music Popularity Classification Using Machine Learning Algorithms. *Electronics* **2022**, *11*, 3518. [CrossRef]
18. Snyder, D.; Garcia-Romero, D.; Sell, G.; Povey, D.; Khudanpur, S. X-Vectors: Robust DNN Embeddings for Speaker Recognition. In Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 15–20 April 2018; pp. 5329–5333. [CrossRef]
19. NIST. Speaker Recognition Evaluation 2016. Available online: https://www.nist.gov/itl/iad/mig/speaker-recognition-evaluation-2016/ (accessed on 30 July 2020).
20. Devi, K.J.; Singh, N.H.; Thongam, K. Automatic speaker recognition from speech signals using self-organizing feature map and hybrid neural network. *Microprocess. Microsyst.* **2020**, *79*, 103264. [CrossRef]
21. Teixeira, F.; Abad, A.; Raj, B.; Trancoso, I. Towards End-to-End Private Automatic Speaker Recognition. *arXiv* **2022**, arXiv:2206.11750.
22. Gao, H.; Hu, M.; Gao, T.; Cheng, R. Robust detection of median filtering based on combined features of the difference image. *Signal Process. Image Commun.* **2019**, *72*, 126–133. [CrossRef]
23. Ma, Z.; Fokoué, E. Accent Recognition for Noisy Audio Signals. *Serdica J. Comput.* **2014**, *8*, 169–182. [CrossRef]
24. Wang, C.; Chen, D.; Hao, L.; Liu, X.; Zeng, Y.; Chen, J.; Zhang, G. Pulmonary image classification based on inception-v3 transfer learning model. *IEEE Access* **2019**, *7*, 146533–146541. [CrossRef]
25. Dong, H.; Xu, Y.; Li, X.; Yang, Z.; Zou, C. An improved ant-lion optimizer with a dynamic random walk and dynamic opposite learning. *Knowl.-Based Syst.* **2021**, *216*, 106752. [CrossRef]
26. Zhang, Y.; Xiong, R.; He, H.; Pecht, M.G. Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries. *IEEE Trans. Veh. Technol.* **2018**, *67*, 5695–5705. [CrossRef]